UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Fernando Luiz de Oliveira

CLUSTERIZAÇÃO DE CONSULTAS EM UM MODELO DE ORDENAÇÃO WEB BASEADO NA RELEVÂNCIA POR TEMPO EM DOMÍNIO ABERTO

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

Raul Sidnei Wazlawick

Florianópolis, agosto de 2005.

CLUSTERIZAÇÃO DE CONSULTAS EM UM MODELO DE ORDENAÇÃO WEB BASEADO NA RELEVÂNCIA POR TEMPO EM DOMÍNIO ABERTO

Fernando Luiz de Oliveira

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação - Área de Concentração Sistemas de Conhecimento, e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

_____

Prof. Dr. Raul Sidnei Wazlawick

Coordenador do Curso

Banca Examinadora

_____

Prof. Dr. Raul Sidnei Wazlawick (Orientador)

_____

Profª. Drª. Renata Vieira

_____

Profª. Drª. Silvia Modesto Nassar

_____

Prof. Dr. Rogério Cid Bastos

“Você não pode ensinar nada a um homem. Você pode apenas ajudá-lo a encontrar a resposta dentro dele mesmo” (Galileu Galilei)

AGRADECIMENTOS

A DEUS, por ter tido força e determinação concluir com êxito mais esta importante fase da minha vida. Agradeço pela saúde e por todo o apoio que tive durante as fases e obstáculos mais difíceis que me foram expostos durante este trabalho.

A minha família, pelas palavras de apoio, de conforto e pela paciência dedicada. Ao meio pai, senhor Guaspar e minha mãe, senhora Maria Rosalina, por ter me dado os fundamentos necessários para aqui chegar. Aos meus irmãos, Anderson, Wesley e Emerson, por ter agüentado, pacientemente, os meus momentos de stress.

Aos meus amigos, Fernanda, Fabiano, Parcilene, Renatto, Francisco Junior, pelas palavras de apoio, pelas cobranças que tanto me incentivaram a buscar forças para concluir este trabalho.

Aos meus amigos de trabalho, em especial a Thereza, que de uma forma ou de outra participaram ativamente desta fase da minha vida.

Ao meu orientador, professor Raul Sidnei, que tanto contribuiu para que este trabalho fosse finalizado, orientando, cobrando e contribuindo para o meu enriquecimento científico e pessoal.

Agradeço a vida, pelas possibilidades e caminhos que me foram permitidos trilhar...

SUMÁRIO

LISTA DE TABELAS

LISTA DE FIGURAS

RESUMO

Com o intuito de aumentar a eficácia e a qualidade dos resultados fornecidos pelas ferramentas de busca, algumas pesquisas vêm buscando aprimorar o processo de ordenação de documentos, ao invés de desenvolver técnicas que foquem apenas na obtenção de informações úteis. Uma destas pesquisas desenvolveu um modelo de ordenação fundamentado na utilização de informações acerca do tempo despendido pelos usuários durante a leitura de cada documento para redefinir a relevância deste documento a uma determinada consulta (*feedback*). No entanto, para que este modelo possa ser implementado em uma ferramenta de busca é necessário que um conjunto de informações sejam armazenadas, tal como a relevância por tempo de cada documento à consulta que a retornou, gerando, desta forma, um grande volume de informação a ser gerenciado. O objetivo deste trabalho é criar as estruturas necessárias para que este modelo seja adotado em uma ferramenta de busca de domínio aberto e comercial. Para tanto, foi criado um metabuscador, no qual o modelo foi implementado, assim como um método de clusterização, responsável por reduzir o custo computacional envolvido.

ABSTRACT

With intention to increase the efficacy and quality of the results obtained by search tools, some researches are investigating how to improve the document ranking process, instead of developing techniques for the acquisition of useful information. One of these researches developed a ranking model based on use of information about the time expended by users during the reading of each document to redefine its relevance in determined query (feedback). However, the implementation of this model in a search tool needs to store information, such as the relevance per time of each document that a query retrieved, generating a large volume of data to be managed. The purpose of this work is to create the necessary structures to adopt this model in an open and commercial search tool. For this, a metasearcher was created, in which the model was implemented, well as a clustering method, responsible for reducing the involved computational cost.

1 INTRODUÇÃO

A *World Wide* Web (WWW) foi desenvolvida, inicialmente, com o intuito de se transformar em um grande repositório do conhecimento humano, possibilitando aos seus colaboradores, mesmo estando distribuídos geograficamente, compartilharem suas idéias e todos os aspectos de um projeto comum (BERNERS-LEE, 1994). Assim, por ter esta finalidade, a mesma foi projetada para fornecer aos seus usuários um ambiente fácil de usar, onde as informações fossem facilmente publicadas e distribuídas. Estas características contribuíram para que a Web se popularizasse, tornando-a uma importante ferramenta para atividades que envolvam a informação.

Desta forma, a Web, em virtude de sua aceitação e disseminação, passou a ter importância equivalente às bibliotecas tradicionais, porém, sem apresentar o principal problema inato a estas, o qual refere-se às dificuldades de acesso por causa da distribuição geográfica (MACEDO, 2001). No entanto, as facilidades advindas com a WWW no processo de distribuição da informação contrastam com sua fragilidade na tarefa de fornecer os mecanismos necessários para promover as devidas formas de se chegar a informações específicas que satisfaçam as necessidades de seus usuários. Este processo de recuperação de informação é incipiente devido a algumas características intrínsecas à Web. Uma destas, refere-se à linguagem usada para estruturar a informação, chamada *HyperText Markup Language* (HTML), que foi desenvolvida para ser simples e ter baixa complexidade. Porém, esta simplicidade não veio acompanhada de uma estrutura que permitisse definir semanticamente as informações, o que dificulta a recuperação de informações relevantes ao usuário devido ao caráter dúbio entre muitas palavras. Outro fato que dificulta a recuperação de informações refere-se à quantidade

de informação que é produzida e disponibilizada, o que torna o trabalho de seleção complexo e demorado.

Neste contexto, aplicam-se os sistemas de Recuperação de Informação (RI), responsáveis pela representação, armazenamento, organização e acesso aos itens de informação (BAEZA-YATES & RIBEIRO-NETO, 1999). Por causa da necessidade da Sociedade da Informação em obter informações que sejam úteis, em um tempo considerado aceitável, estes sistemas de RI ganharam importância, e diversas pesquisas têm sido desenvolvidas em torno deste tema. Uma destas linhas aborda a questão da representação da informação, a qual estuda novas formas de estruturar a informação, levando em consideração a semântica dos dados. Assim, seria possível dotar os sistemas de RI de certa capacidade que lhes permitiriam diferenciar melhor as informações, o que resultaria na recuperação de documentos mais próximos das necessidades do usuário. Outras pesquisas trabalham com a expectativa de que, ao invés de se preocupar somente em recuperar informações úteis, optam por dar ênfase na ordenação dos itens recuperados objetivando, assim, listar os mais importantes nas primeiras posições da lista de resultados. Esta dissertação concentra-se nesta última linha de pesquisa, tal como o trabalho desenvolvido por DETERS (2003).

O trabalho de DETERS (2003) define um modelo de ordenação baseado na combinação entre a freqüência relativa[1] e o tempo de permanência do usuário nos documentos Web, resultando em um valor que especifica o quanto um documento é relevante a uma determinada consulta. A base deste modelo se fundamenta na hipótese de que o conhecimento adquirido acerca do tempo despendido pelos usuários nos documentos possa ser utilizado no processo de *feedback*, ou seja, na redefinição da relevância deste documento a uma determinada consulta. Neste ponto, entenda-se por *feedback* como sendo o mecanismo através do qual um sistema pode aprimorar a qualidade de seu resultado, tendo como base informações passadas (RIJSBERGEN,

[1] Freqüência Relativa: média definida pelo número de ocorrências de uma determinada palavra (Freqüência Absoluta) pela total de palavras existente no documento.

1979). Desta forma, a partir da relevância atribuída à associação entre cada documento a sua respectiva consulta, pode-se reordenar a lista dos documentos recuperados, gerando *rankings*[2] que se modificam de acordo com as informações obtidas durante a navegação dos usuários.

Para validar este modelo, foi desenvolvido por DETERS (2003) um sistema de busca, no qual o modelo foi implantado. Porém, como o objetivo era a validação do modelo e não a ferramenta de busca, foi indexada apenas uma pequena coleção de documentos, referente à Programação Orientada a Objetos, assim como um pequeno número de documentos irrelevantes para este domínio. A ferramenta foi disponibilizada e utilizada por um grupo de acadêmicos, que tinham interesse no domínio utilizado na indexação. Durante o período de validação, foram coletados dados os quais serviram de base para, verificar a eficácia do modelo. Assim, no decorrer de sua utilização, pode-se verificar que os documentos irrelevantes, mesmo tendo sido manipulados de forma a serem listados, inicialmente, nas primeiras posições do *ranking* gerado pela ferramenta, à medida que as consultas foram se repetindo, começaram a decair no *ranking*, dando lugar aos documentos relevantes.

No entanto, por se tratar de uma experiência restrita ao meio acadêmico, algumas questões não foram levadas em consideração durante o seu desenvolvimento. Uma destas questões refere-se à forma como o tempo de permanência do usuário em cada documento visitado seria capturado. No caso do sistema desenvolvido, isto foi solucionado a partir da modificação da estrutura dos documentos, acrescentando uma função para este fim em cada documento que compunha a coleção analisada. Esta solução só foi viável porque todos os documentos analisados se encontravam em um mesmo servidor, tornando possível o acesso ao código-fonte de todos estes documentos. Contudo, esta mesma solução seria inviável para um sistema de busca que trabalhasse com um domínio aberto e com pretensões comerciais, visto que o mesmo não teria

---

[2] *Ranking*: Lista dos documentos recuperados apresentada em ordem decrescente do peso (relevância) atribuída a cada documento para a consulta analisada.

acesso ao código-fonte dos documentos consultados, uma vez que a ferramenta de busca só oferece *links* para estes referidos documentos.

Além da obtenção do tempo de permanência, uma outra lacuna deixada em aberto refere-se ao custo computacional (associado, principalmente, ao número de registros na base de dados) envolvido na manutenção e na recuperação das informações necessárias para a atualização do modelo. Este problema se fundamenta na quantidade de informações a ser armazenada, uma vez que, para manter o modelo, é necessário armazenar tanto as informações sobre consultas, como sobre os *links* retornados, assim como um peso que específica a relevância de cada *link* à consulta que o retornou. Para um domínio específico e com uma coleção de documentos bem definida, tal como foi utilizado no referido trabalho, este conjunto de informações não se configuraria como um problema, visto que o conjunto de URL's e de possíveis consultas é finito. No entanto, em um domínio aberto e comercial, seria difícil sua utilização sem alguma forma de diminuir o custo computacional envolvido. Para exemplificar esta complexidade, suponha que sejam realizadas cem (100) consultas novas em um dia, sendo que para cada uma foram retornados cinqüenta (50) *links*. Como cada *link* estará associado a uma consulta e terá um valor que representa sua relevância a esta consulta, para manter estas informações seriam necessários cerca de cinco mil (100*50) novos registros para armazenar este conjunto de informações. Assim, por causa do grande número de requisições que as diversas ferramentas de busca recebem atualmente, estando em torno de milhões nas ferramentas mais conhecidas, torna-se evidente a necessidade da realização de estudos que possam diminuir este custo computacional, de forma a tornar viável a utilização do modelo.

Esta dissertação aborda a questão do acesso aos itens de informação, bem como o processo de ordenação dos documentos. Porém, centra-se nas lacunas deixadas no trabalho desenvolvido por DETERS (2003), utilizando o modelo desenvolvido e trabalhando com a hipótese de que o mesmo possa ser aplicado e utilizado em um

ambiente comercial de forma equivalente a sistemas de RI conhecidos, tais como o Google[3] e todoBr[4], no processo de recuperação de informações diversas.

Como este trabalho concentra-se, principalmente, na questão da ordenação dos documentos recuperados, o mesmo não desenvolverá todos os processos de recuperação de informação, composto pelo processo de consulta, indexação e ordenação, pois se pressupõe que partes destes processos já estão desenvolvidas e são possíveis de serem usadas. Assim, foi desenvolvido neste trabalho um metabuscador, que é um mecanismo definido na literatura como um sistema de recuperação de informação que trabalha sobre os resultados de outros buscadores. Desta forma, ao invés de se preocupar com a implementação de estruturas e algoritmos de busca e indexação já existentes, pode-se enfatizar o processo de obtenção e manutenção das informações necessárias para a atualização do modelo de ordenação adotado.

Portanto, para tornar este modelo de ordenação de documentos possível de ser implantado e utilizado em uma ferramenta de busca que trabalhe em âmbito comercial, algumas características devem ser analisadas com relação ao custo computacional envolvido. Esta análise faz-se necessária porque os sistemas de RI devem fornecer bons resultados, porém, em um tempo de resposta aceitável e dentro das possibilidades computacionais, pois tanto o processamento quanto a capacidade de armazenamento são finitos. Para viabilizar este modelo, foram desenvolvidos os meios para promover a otimização do custo computacional e, assim, possibilitar que as informações usadas na ordenação (informações acerca dos documentos, das consultas e das relevâncias de tempo destes documentos) possam ser acessadas, sem, contudo, penalizar a ferramenta de busca com tempo excessivo na resposta ao usuário. Para tanto, foi criado um método de clusterização de consultas, o qual se fundamenta na correlação entre as relevâncias por tempo associados ao grupo de documentos em comum à suas respectivas consultas para criar os *clusters*.

---

[3] Google: http://www.google.com.br

[4] todoBr: http://www.todobr.com.br

1.1 Motivações

As facilidades advindas com a Web, com relação à publicação e acesso à informação, contrastam com sua fragilidade, relacionado a localização de informações que satisfaçam as necessidades específicas do usuário. Devido a esta fragilidade e à grande quantidade de informação disponível, torna-se importante que os sistemas de RI forneçam formas para se chegar a tais informações. Assim, este trabalho foi desenvolvido na expectativa de proporcionar os meios necessários para que o modelo de ordenação proposto em DETERS (2003) possa ser utilizado por uma ferramenta de busca que trabalhe com um domínio aberto e comercial.

1.2 Objetivos da Pesquisa

1.2.1 Geral

Este trabalho tem por objetivo expandir o modelo de ordenação proposto por DETERS (2003) de forma a reduzir sua complexidade de espaço para que possa ser implantado em ferramentas de busca que tenham fins comerciais e que não sejam restritas a um domínio específico.

1.2.2 Específicos

Como objetivos específicos deste trabalho, foram abordados os seguintes itens:

a) Utilizar ou desenvolver técnicas/algoritmos que permitam diminuir o custo computacional envolvido para manter o modelo de ordenação atualizado.

b) Melhorar a qualidade dos *rankings* gerados, apresentando os documentos mais relevantes nas primeiras posições.

1.3 Metodologia

Para que os objetivos deste trabalho fossem alcançados foi desenvolvido um metabuscador, o qual utilizou o modelo de ordenação desenvolvido por DETERS

(2003). Para tanto, foram desenvolvidas as estruturas necessárias para manter as informações pertinentes para a manutenção do modelo, levando em consideração o custo computacional envolvido. Assim, para validar o protótipo e, conseqüentemente, o modelo de ordenação, o metabuscador foi disponibilizado na Internet para ser utilizado pelos usuários Web.

A partir da utilização da ferramenta de busca, informações foram geradas e obtidas para atualizar o modelo de ordenação. No entanto, para manter estas informações, tornam-se necessário prover os meios necessários para diminuir o custo computacional envolvido. No caso, entende-se como custo computacional o espaço e o tempo necessário para armazenar, recuperar e atualizar as informações importantes para o modelo de ordenação. Assim, para manter estas informações, foram criadas regras de gerenciamento, as quais são as responsáveis por definirem o momento para atualizar a base de consultas, local onde estas informações estão armazenadas. Porém, somente este gerenciamento não foi suficiente para diminuir o volume de informações armazenadas. Por isto, foi criado um método de clusterização de consultas, baseado na relevância por tempo de cada documento a consulta que o retornou, com objetivo de criar *clusters* de consultas semelhantes, identificadas por um conjunto de documentos relevantes em comum.

Para averiguar a eficácia do método de clusterização criado, foram realizados alguns experimentos com a base de consultas utilizada pelo metabuscador. Os experimentos seguiram a mesma metodologia, porém, se distinguem pelo valor adotado para definir a proximidade entre duas consultas candidatas a participar de um mesmo *cluster* que, no caso, foram utilizados os valores 0.1 e 0.5 para a correlação entre as consultas. Estes experimentos permitiram constatar o quanto o custo computacional (medido em números de registros existentes na base de dados) foi reduzido a partir do método criado, assim como medir a taxa de erro associado à base de consultas que foi submetida ao método de clusterização, se comparada à base de dados que não sofreu a intervenção.

1.4 Justificativa

A Web proporcionou a universalização da informação, porém, a quantidade de documentos existentes dificulta o processo de recuperação destas informações. As ferramentas de busca surgem como uma opção nesta tarefa, mas devido a algumas características particulares ao usuário e ao processo de recuperação, os resultados provenientes destes sistemas de RI não satisfazem inteiramente a necessidade de informação de seus usuários. Estas particularidades são citadas a seguir:

1. A maior parte dos usuários acessa apenas os dez primeiros documentos recuperados (MACEDO, 2001).

2. Elevada recuperação de documentos não-relevantes, devido a:

   2.1. Ambigüidade da língua (palavras idênticas, mas com significados diferentes) e, por isto, muitos documentos são recuperados, mas não são relevantes para determinada consulta (MOSTAFA, 2002).

   2.2. Manipulação dos documentos, onde estes são previamente elaborados para aparecer nos resultados das consultas.

O segundo item (citado acima) provoca o inchaço da lista de resultados (*ranking*), o que faz com que, na maior parte das vezes, muitos documentos relevantes não sejam vistos como conseqüência do primeiro item. Assim, é importante desenvolver técnicas que privilegiem os documentos relevantes[5], fazendo-os aparecer nas primeiras posições do *ranking*. O trabalho desenvolvido por DETERS (2003) propôs um modelo de ordenação com este objetivo. Porém, o mesmo apresenta algumas limitações, tais como:

[5] Espera-se que um documento que seja lido e que tenha uma relevância de tempo alta seja considerado relevante pelo usuário.

1. Foi desenvolvido e testado de forma limitada sobre um conjunto relativamente pequeno de documentos sobre Programação Orientada a Objetos.

2. Por ter um domínio de pesquisa bem definido, o conjunto de consultas possíveis de serem realizadas é bem definido e relativamente pequeno. Porém, em um domínio aberto, o custo computacional cresce de acordo com o aumento do número de consultas realizadas.

Para tornar o modelo viável torna-se necessário prover as soluções para suprir as limitações citadas acima, resultado em uma ferramenta de busca que abranja uma parte maior da Web e que possa fornecer melhores resultados.

1.5 Resultados Esperados

Espera-se que a partir dos resultados deste trabalho seja possível construir sistemas de busca com *rankings* de documentos mais adequados às necessidades dos usuários, ou seja, com maior recuperação de documentos relevantes e menor recuperação de documentos irrelevantes.

1.6 Limitações do Trabalho

Este trabalho dará continuidade ao de DETERS (2003), porém, como não focará a resolução de todas as limitações apresentadas por este, algumas de suas limitações não serão exploradas neste trabalho, tais como:

a) Pré-definição do número de palavras que um usuário qualquer lê por segundo, onde será assumido o valor de três (3) palavras lidas por segundo (BOCK, 2001).

b) Pré-definição (sem um estudo apropriado) do limite inferior e superior do tempo de permanência do usuário na página. Isto se verifica pela necessidade de evitar situações na qual o usuário ou tenha esquecido uma página aberta ou tenha aberto, mas não lido.

c) Não será levada em consideração a diferença entre o tempo de leitura de um documento que contenha imagens de um que não contenha. Assume-se que todas as páginas são compostas por textos e que o tempo de permanência esperado para a leitura desta página seja definido apenas pelo número de palavras presente na mesma.

Além destas, inerentes ao trabalho original, outras limitações são assumidas, tais como:

a) Não serão realizadas avaliações para verificar a eficácia do *ranking* gerado a partir da expansão do modelo de ordenação.

b) Tempo de resposta maior do que outras ferramentas de busca, pois como este trabalho foi implementado como um metabuscador, o tempo resultante é a soma do tempo de busca no sistema de RI fonte, com o acréscimo do tempo na reordenação dos documentos usando o modelo proposto.

c) Como os documentos recuperados não estão no mesmo servidor e, por isto, os códigos-fonte não estão disponíveis para que seja acrescentada a função para capturar o tempo de permanência, é necessário criar alternativas para que este tempo seja obtido. Portanto, como não há um mecanismo apropriado para capturar o tempo inicial e final do acesso do usuário ao documento, possivelmente, este tempo é menos preciso do que o capturado no trabalho de DETERS (2003).

1.7 Estrutura do Trabalho

Este trabalho está organizado da seguinte forma: as seções 2, 3, 4 e 5 apresentam os conceitos-chave para a compreensão do que foi desenvolvido neste trabalho. Para tanto, a seção 2 apresenta os conceitos referentes à recuperação de informação, assim como a classificação das diversas ferramentas de busca Web. A seção 3 aborda os critérios utilizados para compor os modelos de ordenação, estruturas responsáveis por montar os *rankings*. Já a seção 4 foca os conceitos que envolvem o processo de clusterização de documentos. As demais seções apresentam o que foi desenvolvido, onde, na seção 5 é

apresentada a ferramenta de busca desenvolvida, assim como as estruturas necessárias para torná-la viável. Finalmente, a seção 6 aborda as considerações finais deste trabalho.

2 RECUPERAÇÃO DE INFORMAÇÃO

Segundo MOEERS (1951), a recuperação de informação pode ser definida como sendo um processo ou método onde o usuário converte a sua necessidade de informação em uma lista real de documentos que contenham informações relevantes a sua consulta. Afirma ainda que, a recuperação de informação abrange os aspectos intelectuais da descrição da informação, a sua especificação para a busca, assim como a utilização de sistemas ou máquinas no processo de recuperação.

O problema do armazenamento e da recuperação de informação passou a ser objeto de pesquisas desde o instante no qual a informação teve seu papel alterado, passando a ter importância estratégica perante a Sociedade da Informação. A conseqüência direta desta alteração foi o aumento significativo da quantidade de informação produzida. Em decorrência, outros problemas surgiram, tal como as dificuldades referentes ao acesso a informações específicas, o que torna necessário o desenvolvimento de estruturas que auxiliem o acesso a estes itens de informação.

Segundo RIJSBERGEN (1979), a princípio, o armazenamento e a recuperação de informação é uma tarefa simples. Supõe-se, inicialmente, que exista um repositório de documentos e uma pessoa (usuário deste repositório) capaz de formular consultas que representem sua necessidade de informação. A aplicação destas consultas resultaria em um conjunto de documentos que satisfaria sua necessidade específica de informação. Este usuário poderia obter este conjunto de documentos relevantes, simplesmente, lendo todos os documentos do repositório, guardando os relevantes e descartando os demais. Este seria o processo de recuperação que forneceria o melhor resultado. Porém, em um ambiente Web, esta seria uma solução impraticável devido ao volume de informação

existente. O usuário não teria tempo e nem gostaria de dedicá-lo a tarefa de ler toda a coleção, o que torna esta saída inviável e justifica a necessidade dos Sistemas de Recuperação de Informação.

2.1 Sistemas de Recuperação de Informação

Com o crescimento do volume de documentos disponibilizados para os usuários através da rede mundial de computadores e devido a enorme diversidade de conteúdos encontrado nos mesmos, tornou-se necessário o desenvolvimento de sistemas de RI Web. Uma das maiores dificuldades encontradas por estes sistemas refere-se à questão da recuperação rápida e precisa de documentos relevantes às necessidades do usuário.

Com a Web, a dependência por sistemas para auxiliar o processo de recuperação de informação ficou mais evidente, visto que esta proporcionou alterações significativas com relação à forma como a informação é publicada, gerando excessos de informações e dificultando o processo de RI útil para seus usuários. No entanto, apesar do acesso rápido e eficiente à informação ser um problema cuja origem é bem anterior a sua existência e do próprio computador, as ferramentas desenvolvidas e disponíveis facilitaram, mas não resolveram o problema da recuperação eficiente da informação. O efeito direto deste problema está relacionado com o fato de que informações relevantes, por não serem recuperadas, são replicadas, resultando em desperdício de tempo e esforço. Desta forma, existe uma constante necessidade por pesquisas que visem desenvolver sistemas de RI que auxiliem e aprimorem as atividades que envolvem a representação, o armazenamento e a recuperação da informação.

O problema da RI pode ser caracterizado da seguinte forma: “como distinguir uma informação relevante de uma irrelevante”. Os sistemas automáticos que visam solucionar este problema são chamados de sistemas de recuperação de informações (HUIBERS & LAMAS & RIJSBERGEN, 1996). Estes sistemas visam armazenar, manter e recuperar informações, sendo estas estruturadas ou não. Seu objetivo principal é fornecer os meios que permita diminuir o esforço humano na busca de informações, recuperando informações que realmente atendam as necessidades dos usuários. Para realizar estas atividades, implementam um conjunto de estruturas, as quais compõem o processo de recuperação de informação, que será apresentado na seção seguinte.

2.2 Processo de Recuperação de Informação

Segundo HUIBERS & LAMAS & RIJSBERGEN (1996) a eficiência de um SRI Web está diretamente relacionada a dois parâmetros: abrangência e precisão dos resultados. A abrangência refere-se ao poder de análise de diferentes tipos de documentos utilizando a mesma estrutura de indexação. Já a precisão dos resultados muitas vezes está diretamente relacionada com questões referentes a ambigüidades inerentes à língua padrão, dificuldades do usuário na elaboração das consultas, diferenças entre os vocabulários do autor do documento e do usuário do sistema de recuperação, assim como deficiências no processo de indexação. Tanto o primeiro parâmetro, como o segundo estão diretamente relacionados ao processo de recuperação de informação de um sistema de busca, e sofrem influência conforme estes processos são desenvolvidos. A Figura 1 apresenta o processo de recuperação de informação comum a maioria dos sistemas de busca.

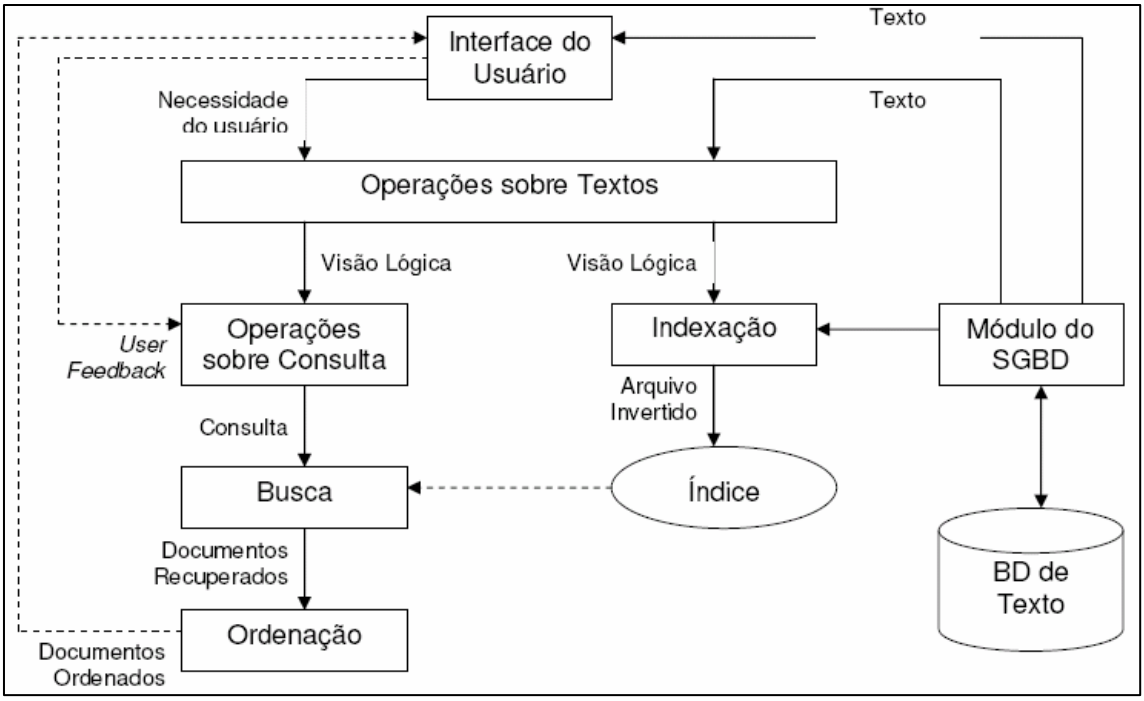

Figura 1 – Processo de Recuperação de Informação: Modificado de (BAEZA-YATES & RIBEIRO-NETO, 1999)

Apesar de não estar explícito no esquema apresentado na Figura 1, implicitamente o processo de recuperação de informação pode ser dividido em:

processo de consulta, processo de ordenação e processo de indexação, que serão apresentados a seguir:

- O processo de consulta se inicia com o usuário descrevendo sua necessidade de informação, usando para isto a interface do sistema. Porém, antes de executar a consulta, algumas outras operações são realizadas. Uma destas refere-se ao tratamento do texto, extraindo deste, palavras que não são úteis para a pesquisa, tais como: artigos, preposições, conjunções, entre outras. Posteriormente, são realizadas operações de consulta, onde estes termos são associados a operadores lógicos (*and*, *or* ou *not*), a fim de aumentar a precisão e eficiência da busca. Após estas atividades a consulta é executada, e uma lista de documentos é obtida.

- Tendo esta lista de documentos, inicia-se o trabalho da ordenação (*ranking*) dos documentos, cuja responsabilidade cabe ao processo de ordenação. Para tanto, modelos de ordenação são construídos e cada sistema de busca possui o seu próprio modelo, baseado em critérios que permitam melhor identificar a relevância de cada documento recuperado à consulta do usuário. Como resultado deste processo obtém-se um *ranking* dos documentos (índices) recuperados, o qual será apresentado ao usuário.

- Por fim, o processo de indexação, que visa construir as estruturas de índices sobre os quais o processo de consulta realizará as pesquisas. Este processo fundamenta-se sobre duas estruturas: repositório de documentos e indexação. A primeira representa a coleção de documentos a ser indexada. Já a segunda refere-se ao processo de criação dos índices propriamente dito, onde, a princípio, índice pode ser visto como sendo o conjunto de termos que melhor identificam um determinado documento, além das informações sobre o documento em si, como, por exemplo, seu endereço Web.

Os sistemas de RI Web se classificam de acordo com a forma como implementam ou não os processos descritos acima. Esta classificação, bem como as características existentes em cada tipo de sistema de busca, será o foco da próxima seção.

2.3 Classificação dos Sistemas de Recuperação de Informação Web

Esta seção apresenta os diversos sistemas de busca existentes na Web. Porém, por não ser o foco deste trabalho fazer um estudo detalhado sobre o funcionamento de cada um, os mesmos serão brevemente descritos, enfatizando apenas os mecanismos de busca e os metabuscadores nos quais este trabalho se fundamenta para implementar os processos de RI, tais como descritos na seção 2.2.

2.3.1 Mecanismos de Busca

Mecanismo de busca corresponde a uma ferramenta assistida por um conjunto de artefatos de software, responsáveis por automatizar todas as etapas que envolvem o processo de recuperação de informação. Estes artefatos estão representados na arquitetura definida por ARASU (2001), apresentada a seguir:

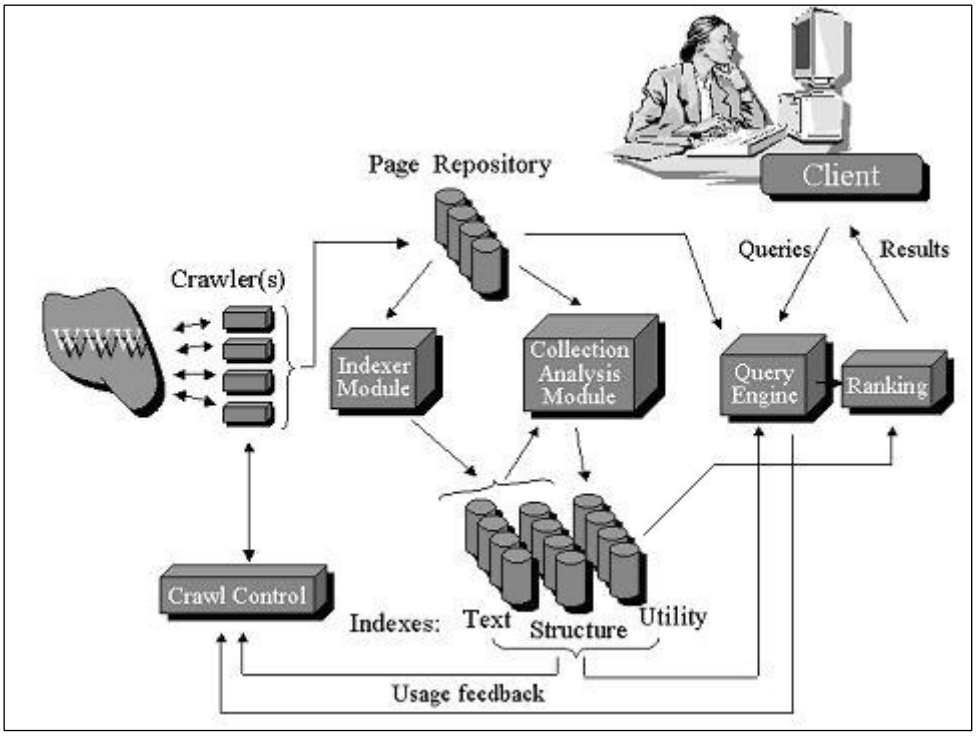

Figura 2 – Arquitetura de um Mecanismo de Busca (ARASU, 2001)

Esta arquitetura representa um mecanismo de busca completo e que engloba todas as etapas referentes ao processo de representação, armazenamento e recuperação da informação. Esta arquitetura é composta basicamente por: *Crawler, Crawl Control, Page Repository, Indexer Module, Collection Analysis Module*, estruturas responsáveis pela indexação e representação dos documentos; *Query Engine*, que visa possibilitar a

tradução das necessidades de informação do usuário em estruturas formais para que um computador entenda; e, *Ranking,* responsável pela ordenação dos resultados provenientes da execução da consulta. A seguir será apresentada a definição de cada uma destas estruturas:

a) *Crawler*: são pequenos programas de apoio aos sistemas de RI que percorrem a Web em busca de documentos, recuperando-os e armazenando-os em uma base, conhecida como repositório de documentos (*Page Repository*). Além do documento em si, outras informações também são coletadas e armazenadas, tal como a *Uniform Resource Locator* (URL) da página recuperada.

b) *Crawl Control*: responsável por coordenar as ações dos *Crawlers*, definindo quais *links* serão visitados a cada varredura, organizando e armazenando as informações provenientes deste processo.

c) *Page Repository*: representa o local onde os documentos recuperados, após a ação dos *Crawlers,* são armazenados.

d) *Indexer Module*: cria uma estrutura de índices, a partir dos documentos armazenados no *Page Repository*, contendo informações que permitem ao *Collection Analysis Module* e ao *Query Engine* recuperar itens (referências) de documentos úteis para uma determinada consulta de um usuário. As principais informações coletadas são: conjunto de palavras que melhor identificam o documento, URL, data de acesso, entre outras.

e) *Collection Analysis Module*: este módulo possui a responsabilidade de criar uma variedade de índices que possam ser úteis para prover o melhor acesso às páginas.

f) *Query Engine*: responsável por receber as consultas dos usuários e realizar a busca.

g) *Ranking*: a partir de alguns critérios de ordenação, este módulo fica com a responsabilidade de organizar o resultado em uma lista, posicionando os documentos mais relevantes à consulta do usuário nas primeiras posições.

Esta arquitetura, dentro do processo de recuperação de informação, pode ser assim contextualizada: o processo de indexação é composto pelas seguintes estruturas: *Crawler, Crawl Control, Page Repository, Indexer Module, Collection Analysis Module*; o processo de consulta é representado pela *Query Engine* e pela interface de consulta propriamente dita; e, por fim, o processo de ordenação, representado pelo *ranking*.

2.3.2 Diretórios

Os diretórios foram os primeiros sistemas de busca desenvolvidos e utilizados na Web, que se caracterizam pela forma como indexam e organizam seus documentos. Esta organização é baseada na forma como os bibliotecários trabalham, ou seja, através de classificações hierárquicas do conhecimento humano (BAEZA-YATES & RIBEIRO-NETO, 1999). Assim, ao contrário dos mecanismos de busca, os primeiros diretórios buscavam e indexavam seus documentos Web com o auxílio de pessoas, e não por meio de componentes de software. Estas pessoas, em sua maior parte composta por bibliotecários, são os responsáveis por catalogar e organizar os documentos Web. Como resultado desta atividade, obtêm-se uma estrutura hierárquica, teoricamente mais simples e intuitiva para a navegação do usuário, composta basicamente por documentos dos quais se acreditam que sejam de melhor qualidade, uma vez que todos os documentos indexados passam pelo crivo de um ou mais bibliotecários.

Porém, atualmente muitos diretórios já contam com sistemas mais automáticos para fazer esta catalogação. Para isto, estes diretórios utilizam técnicas de agrupamento de documentos (*clusters*), baseado em características (conteúdo, estrutura e navegação) dos próprios documentos Web. Isto passou a ser necessário por causa da velocidade com que a Web cresce, o que torna a demanda muito maior do que a capacidade de catalogação manual.

A arquitetura de um diretório é muito parecida com a de um mecanismo de busca, porém, sem a presença dos componentes referentes à indexação automática dos documentos Web. Desta forma, a atualização da base de índices (metadados dos documentos verdadeiros) é feita ou através da navegação dos responsáveis, ou através

dos próprios desenvolvedores, que submetem seus documentos para que possam ser apreciados e catalogados pelos bibliotecários.

Com o intuito de aumentar a abrangência dos resultados oferecidos, muitos diretórios estão desenvolvendo ou utilizando as estruturas referentes a um mecanismo de busca, e vice versa, resultando nos sistemas híbridos. Estes sistemas representam uma tendência utilizada nos ferramentas de busca atuais, agregando a abrangência de um mecanismo de busca, com a qualidade associada aos diretórios.

2.3.3 Metabuscadores

"Metabuscador" é um termo usualmente aplicado a técnicas que combinam os resultados provenientes de outros sistemas de busca, ordenando e aprimorando a recuperação (MONTAGUE & ASLAM, 2002). Desta forma, estas ferramentas podem aumentar a abrangência de suas pesquisas, oferecendo aos seus usuários uma interface comum e padronizada, na qual eles podem escolher entre quais e quantos outros sistemas serão usados para as buscas. Estas características configuram-se como as principais vantagens dos metabuscadores, uma vez que estas ferramentas se transformam em uma camada intermediária, livrando o usuário da necessidade de conhecer os detalhes e configurações de cada ferramenta consultada (BAEZA-YATES & RIBEIRO-NETO, 1999).

Porém, como cada sistema de busca consultado retorna uma lista de documentos, é necessário que este tipo de ferramenta desenvolva técnicas e/ou métodos que permitam eliminar os resultados duplicados e combinar os *rankings* retornados por estas diferentes fontes (KOBAYASHI & TAKEDA, 2000). A forma como esta tarefa é realizada depende dos objetivos de cada metabuscador, onde alguns apenas visam aumentar a abrangência de pesquisa e oferecer uma interface padronizada para as consultas. No entanto, outros, além de oferecer uma maior abrangência, visam melhorar o *ranking* gerado. Para isto, faz-se necessário que os metabuscadores desenvolvam seus próprios modelos de ordenação. Estes modelos são baseados em um conjunto de critérios que auxiliam o sistema a identificar e a classificar os documentos segundo sua relevância à consulta realizada.

Segundo SMITH & HURSON (2003), um metabuscador é composto por quatro (4) unidades funcionais: descoberta de recursos, extração de informação, armazenamento de informações e recuperação de informação. Os três primeiros são desenvolvidos com o auxílio da interação com o usuário e são importantes para o processo de atualização do modelo de ordenação do metabuscador. Já o último aborda todos os mecanismos necessários para manter os primeiros, envolvendo a pesquisa nos demais sistemas de busca, os agentes de interface para capturar as informações da interação com o usuário e a apresentação do resultado.

Fazendo um paralelo entre a arquitetura de um mecanismo de busca com a de um metabuscador, verifica-se que os únicos componentes que são implementados referem-se a interface, o *Query Engine* e *ranking*, estruturas estas que compõem o módulo de consulta do processo de recuperação de informação. No entanto, o metabuscador, caso possua seu próprio modelo de ordenação, necessita armazenar informações pertinentes e necessárias para manter o seu modelo atualizado. Assim, em alguns casos, seria necessário desenvolver outras estruturas que permitam fazer análises de *links* ou análises de documentos, conforme as necessidades de informação do modelo de ordenação.

Contudo, como citado anteriormente, um metabuscador pode optar por criar seu próprio modelo de ordenação, assim como as demais ferramentas de busca. Estes modelos de ordenação são baseados em critérios que auxiliam a ferramenta a distinguir e classificar documentos que poderiam ser relevantes à consulta do usuário, critérios estes que serão abordados na seção seguinte.

2.4 Critérios para Ordenação de Documentos

Um sistema de RI Web processa diversas consultas diariamente, sendo que, para cada uma, é montada uma lista (*ranking*) contendo os resultados. Este *ranking* pode conter documentos úteis ou não à necessidade de informação do usuário. Esta necessidade de informação é representada por meio das palavras-chave definidas pelo próprio usuário. Porém, devido ao caráter dúbio das palavras, e por causa da pouca semântica (significado) existente nos documentos Web, por vezes, o sistema de RI Web recupera informações que não condizem com a real necessidade do usuário. Por causa da

complexidade envolvida no processo de se recuperar apenas documentos úteis, utilizam-se critérios de ordenação na expectativa que, dos documentos recuperados, os mais importantes sejam listados nas primeiras posições do *ranking*.

Para gerar os *rankings* faz-se necessária a utilização de critérios que permitam classificar um documento em relevante ou não a uma determinada consulta. Tendo como base estes critérios, cada ferramenta de busca constrói seu próprio modelo, que pode tanto auxiliar na recuperação dos documentos, como na sua ordenação. A seguir serão apresentados alguns destes critérios.

2.4.1 Critérios baseados em palavras-chave

Inicialmente os modelos de ordenação eram baseados, quase que exclusivamente, nos critérios baseados em palavras-chave. Estes critérios fundamentam-se na hipótese de que o usuário, ao criar um documento, utiliza e distribui certas palavras que podem ser utilizadas para identificar este documento. Assim, através da análise e comparação deste documento com os termos que compõem a consulta, pode-se atribuir pesos a cada documento e, conseqüentemente, ordená-los. Os principais critérios que fazem parte deste grupo são:

a) Freqüência Absoluta: o peso de cada documento é gerado contando-se quantas vezes os termos que compõem a consulta se repetem no conteúdo de cada um dos documentos avaliados.

b) Freqüência Relativa: o peso é gerado fazendo a média entre as vezes que os termos que compõem a consulta se repetem no documento (freqüência absoluta) pelo total de palavras existentes neste documento.

c) Localização das palavras-chave: os pesos são gerados atribuindo valores diferenciados de acordo com a localização da palavra-chave. Desta forma, acredita-se que um documento possa ser mais relevante caso a palavra-chave apareça no título do documento, ou esteja destacado no texto, ou, então, esteja nas primeiras linhas.

Porém, à medida que se conhecia a forma como cada ferramenta de busca construía seus *rankings*, alguns usuários passaram a burlar seus documentos para que os

mesmos fossem bem classificados em uma consulta, ou até mesmo, fossem listados em consultas as quais não seriam relevantes. Isto é possível por causa do caráter frágil dos modelos que trabalhavam exclusivamente com estes critérios, uma vez que todos permitem que o usuário manipule seus documentos de forma a serem bem classificados. Por exemplo, a freqüência absoluta e a freqüência relativa podem ser manipuladas sendo necessário apenas acrescentar possíveis palavras-chave no documento camufladas com a mesma cor do fundo da página. Assim, por serem fáceis de manipular, aos poucos estes critérios passaram a ser utilizados como auxiliares a modelos fundamentados em critérios mais complexos.

2.4.2 Análise de *Links*

Por causa da fragilidade dos modelos baseados em palavras-chave, outras vertentes começaram a ser desenvolvidas. Uma destas vertentes trabalha com a hipótese de que a estrutura de *hyperlink* pode gerar medidas que auxiliem na definição da relevância de um documento, além de auxiliar as técnicas de clusterização na classificação dos documentos relevantes (KOBAYASHI & TAKEDA, 2000). Através desta abordagem diversos algoritmos foram criados, sendo que entre estes, destaca-se o *PageRank*, utilizado no modelo de ordenação do mecanismo de busca Google.

O *PageRank* de um documento Web é calculado somado todos os valores (importância) atribuídos aos documentos que apontam para ele. Esta importância define o quanto um documento (v) é importante para outro (u), sendo que o valor a ser atribuído a esta 'importância' de 'v' para 'u' é proporcional à 'importância' dada a 'u' e inversamente proporcional ao número de documentos (páginas) que apontam para o mesmo (KAMVAR, 2003). Por este princípio, evita-se que o modelo baseado em *PageRank* possa ser manipulado, visto que a relevância de um documento leva em consideração a relevância dos outros. No entanto, este critério, por se basear na análise de *links*, pode prejudicar um documento recém criado que poderia ser relevante e, conseqüentemente, que poderia estar bem classificado para a consulta do usuário, mas que, por ainda não ser popular, será depreciado no *ranking* gerado.

Desta forma, torna-se necessário associar o critério de análise de *links* a outros para a composição do modelo de ordenação, utilizando suas vantagens, porém, sem

prejudicar os documentos que ainda não são populares. Através desta associação, é possível melhorar os *rankings* gerados, tornando os modelos menos suscetíveis à manipulação.

2.4.3 Análise do comportamento do usuário

Este critério trabalha com a hipótese de que dados oriundos da navegação dos usuários entre os documentos Web podem fornecer informações relevantes, tanto sobre as preferências dos usuários, como da relevância destes documentos a determinadas consultas. Um destes comportamentos refere-se aos dados obtidos a partir dos cliques (*Clickthrough data*), definidos através da tripla (q, r, c) (JOACHIMS, 2002). Esta tripla consiste de uma consulta 'q', o *ranking* apresentado ao usuário (r) e de um conjunto 'c' de *links* clicado pelo usuário. Outro exemplo seria a utilização do tempo de permanência do usuário nos documentos Web, que poderia fornecer informações importantes sobre a relevância do conteúdo à consulta realizada (DETERS, 2003).

Como este critério se fundamenta nas informações obtidas através da navegação do usuário pelos documentos Web, é necessário definir estratégias que permitam com que estas informações sejam capturadas e armazenadas para que o modelo possa ser atualizado. Tendo-se estas informações, podem-se construir modelos de ordenação mais flexíveis, modificando o *ranking* gerado para uma determinada consulta sempre que o perfil dos usuários se modificar.

3 MODELO DE ORDENAÇÃO

Inicialmente, muitos sistemas de busca foram criados, e estes se adaptaram bem com as técnicas e algoritmos de recuperação de informação existentes (ARASU, 2001). No entanto, estes algoritmos de RI foram desenvolvidos para coleções de documentos relativamente pequenas e coerentes, tais como artigos de notícias e catálogos de livros em uma biblioteca. A Web, por outro lado, caracteriza-se por possuir grandes coleções de documentos, com baixa coerência e com elevado grau de mudanças. Assim, para que os sistemas de RI pudessem trabalhar de forma eficiente na Web foi necessário que novas técnicas, ou extensões das anteriores, fossem criadas. Estas técnicas precisam ser capazes de construir estruturas de índices escaláveis e eficientes, além de melhorar a capacidade dos sistemas de busca em discriminar documentos relevantes dos não relevantes (ARASU, 2001).

Uma pessoa, do ponto de vista intelectual, é capaz de estabelecer a relevância de um documento a sua necessidade de informação. No entanto, para o computador esta seria uma tarefa que envolveria elevada complexidade, visto sua inabilidade em discernir os documentos relevantes dos que não são. Com o objetivo de dotar um sistema de computação com tal habilidade, torna-se necessário construir modelos capazes de definir e ordenar estes documentos, segundo critérios de relevância (RIJSBERGEN, 1979). Desta forma, vários modelos foram propostos e utilizados nos sistemas de busca e, por isto, existe uma grande diferença entre os resultados apresentados por estes diversos sistemas. A seção seguinte apresenta os estudos referentes a um destes modelos.

3.1 Modelo de ordenação desenvolvido por DETERS (2003)

O trabalho desenvolvido em DETERS (2003) criou um modelo de ordenação que permite agregar o conhecimento adquirido acerca do tempo despendido pelos usuários nos documentos Web no processo de ordenação destes documentos. Assim, cada vez que uma determinada consulta é repetida, as novas informações adquiridas servem de base para a atualização do modelo e, conseqüentemente, para a modificação da classificação de cada documento no *ranking* gerado. Estas informações, obtidas a partir da análise do comportamento do usuário, auxiliam o modelo de ordenação a retirar o caráter estático dos *rankings* que se fundamentam apenas nos critérios baseados em palavras-chave. Desta forma, o conjunto de fórmulas matemáticas que compõem o modelo possibilita com que cada documento Web retornado na execução de uma consulta modifique sua classificação, conforme a importância que os usuários atribuírem ao documento, importância esta definida pela leitura ou não do documento.

Além da relevância por tempo, o modelo utiliza a freqüência relativa das palavras-chave presentes na consulta, combinando-os para gerar o peso de cada documento à consulta. Porém, o modelo de ordenação somente pode ser aplicado após a primeira ocorrência de uma determinada consulta, uma vez que, para seu funcionamento e atualização é necessário obter o tempo de permanência do usuário em cada documento retornado nesta consulta. Assim, a princípio, apenas parte do modelo é utilizado, mais precisamente o critério da freqüência relativa, que fornece suporte para a ordenação inicial dos documentos. Uma vez tendo as informações necessárias para a aplicação do modelo, a relevância de cada documento à consulta é atualizada e armazenada, servindo de base para a ordenação destes documentos na próxima vez que esta referida consulta ocorrer.

Contudo, para que o modelo pudesse ser testado e validado, foi desenvolvido um mecanismo de busca parcial, sendo que, da arquitetura geral apresentada na seção 2.3.1, foram implementados apenas os módulos referentes à consulta, a geração do *ranking* e parte do módulo de indexação. Para efeito de teste, foi criada para este último componente apenas uma estrutura de índices, baseada em uma coleção de documentos referentes à Programação Orientada a Objetos. A Figura 3 apresenta a arquitetura do sistema desenvolvido por DETERS (2003).

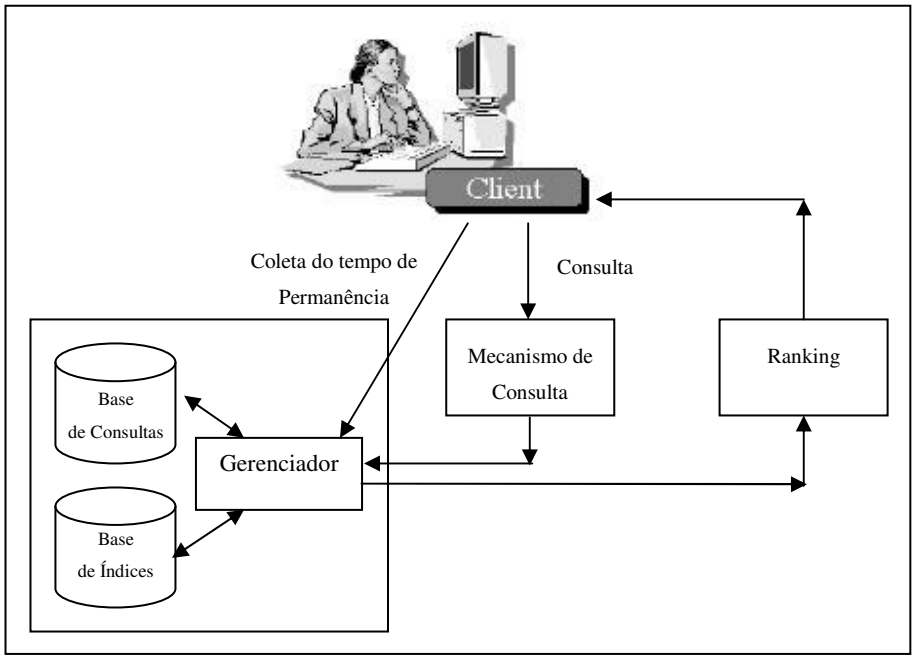

Figura 3 – Arquitetura do sistema de busca desenvolvido em DETERS (2003)

Nesta arquitetura, o processo de recuperação de informação começa com o usuário especificando suas necessidades de informação, via interface da ferramenta. O mecanismo de consulta modifica este texto, adaptando-o para que possa ser entendido pelo sistema, retirando os termos pouco representativos, tais como: artigos, preposição, conjunção, entre outros. Os termos resultantes são repassados para o módulo de gerenciamento, responsável por executar a consulta. Este módulo verifica na base de consultas se já houve alguma ocorrência da consulta atual, onde: caso tenha, os itens de informação referente a esta consulta são recuperados e ordenados segundo a relevância por tempo; caso não, a busca é realizada tendo como base o conjunto de palavras-chaves, utilizando a freqüência relativa para fazer a ordenação da lista resultante. A etapa final deste processo refere-se ao monitoramento do usuário, sendo esta a etapa responsável por fazer a coleta dos tempos de permanência do usuário em cada documento consultado. Estas informações obtidas auxiliam no *feedback* desta consulta, sendo utilizadas para atualizar o modelo.

3.1.1 Funcionamento do Modelo

Como já foi mencionado, o modelo desenvolvido utiliza dois critérios no processo de ordenação dos documentos. O primeiro refere-se à freqüência relativa, responsável por fazer a ordenação preliminar dos documentos recuperados. Já o segundo (foco deste trabalho) refere-se à relevância por tempo, que têm o objetivo de agregar informações obtidas a partir do usuário, no caso, o tempo de permanência destes nos documentos, ao processo de ordenação. Estes critérios combinados compõem o peso final (relevância) de cada documento perante a consulta, originando o *ranking* final dos documentos recuperados.

Assim, dentro do contexto do modelo, a freqüência relativa é a responsável por atribuir um peso fixo à relevância de cada documento e a relevância por tempo, um valor variável, de acordo com a avaliação dos usuários da ferramenta. Isto fornece a mobilidade e a possibilidade para que um documento mude sua classificação nos *rankings*. Desta forma, por ser centrado na importância que os usuários atribuem aos documentos retornados para a consulta, trabalha-se com a hipótese de que um documento é relevante para este usuário, caso este leia o documento. A partir desta hipótese, três situações são esperadas:

a) O usuário ficou mais do que o mínimo do tempo esperado para a leitura do documento e, portanto, terá sua relevância aumentada para a consulta em questão.

b) O usuário ficou somente o mínimo esperado e sua relevância ficará inalterada.

c) O usuário não leu o documento (o tempo de sua permanência ficou abaixo de um terço (1/3) do tempo esperado para a leitura do documento) e, assim, terá sua relevância diminuída.

À medida que as consultas forem repetidas, acredita-se que o *ranking* gerado tenderá a ter uma melhor qualidade, isto porque os documentos irrelevantes tenderão a decair na lista de resultado, ficando nas primeiras posições apenas os itens relevantes para a consulta em questão. A outra vantagem do modelo refere-se a sua capacidade em tornar os *rankings* menos suscetível à manipulação dos desenvolvedores, visto que

mesmo que o documento tenha sido manipulado para aparecer bem classificado no *ranking*, ele tenderá a cair à medida que ele não for aceito (lido) pelos usuários.

No entanto, para que o modelo possa ser atualizado, torna-se necessário obter e armazenar um conjunto de informações, tais como: palavras-chave da consulta, a lista de documentos, o tempo esperado de leitura para cada documento, a relevância por tempo e a freqüência relativa. Assim, a cada consulta, o seguinte algoritmo é realizado (Quadro 1):

Quadro 1: Algoritmo de Pesquisa

Entrada: consulta (K)

Procedimentos:

1. SE **K** não existe na Base de Consultas ENTÃO
   1.1. Realizar consulta na Base de Índices;
      1.1.1. Ordenar os documentos pela freqüência relativa;
   1.2. Coletar o Tempo de Permanência do usuário em cada documento;
      1.2.1. Calcular a Relevância por Tempo de cada documento acessado;
      1.2.2. Armazenar as informações na Base de Consultas;
2. SENÃO
   2.1. Realizar consulta na Base de Consultas;
      2.1.1. Ordenar e combinar a freqüência relativa os documentos segundo a Relevância de cada documento à consulta;
   2.2. Coletar o Tempo de Permanência do usuário em cada documentos;
      2.2.1. Atualizar a Relevância por Tempo de cada documento acessado;

Este algoritmo especifica onde a consulta será realizada, assim como a obtenção e armazenamentos das novas informações, que serão utilizadas para uma posterior atualização da base de consultas. Desta forma, o primeiro passo é verificar se a consulta executada já foi realizada por algum outro usuário. Para isto será necessário fazer uma busca pela consulta na base de consultas, sendo que, caso a mesma já tenha sido realizada, a mesma somente será recuperada e ordenada segundo o peso resultante da

associação da relevância por tempo e da freqüência relativa. Caso não, a consulta é executada na base de índices e o *ranking* será gerado levando em consideração apenas a freqüência relativa obtida. À medida que o usuário acessar os documentos retornados, novos dados serão coletados e utilizados para atualizar a base de consultas, tarefa esta que será apresentada na seção seguinte.

3.1.2 Atualização do Modelo

Dado uma consulta Q, um documento d_j e o tempo de permanência de um usuário neste documento, o processo de atualização da relevância por tempo pode ser descrito através do seguinte algoritmo (Quadro 2):

Quadro 2: Algoritmo de Atualização do Modelo

Entrada:

- consulta (K);
- Tempo de Permanência do usuário (u) em um documento (d_j) ($T_{permanência}(u, d_j, k)$);
- Tempo Esperado para a leitura do documento ($T_{esperado}(d_j)$);
- Constante ρ, que por padrão assume o valor de 2;

Procedimentos:

1. SE $T_{permanência}(u, d_j, k) \geq T_{esperado}(d_j) / \rho$ ENTÃO:
   1.1. Calcular a relevância do documento à consulta: $\Delta_1 = \min\{1, (\rho * T_{permanência}(u, d_j, k) - T_{esperado}(d_j)) / T_{permanência}(u, d_j, K)\}$;
   1.2. Calcular o quanto a relevância do documento poderá crescer: $\Delta_2 = \lambda - relevância_{Tempo}$, onde λ, por padrão, pode variar dentro do intervalo [0, 3] e *relevância* $_{Tempo}$ representa a relevância por tempo atual do documento avaliado para a consulta em questão;
   1.3. Atualizar a Relevância por Tempo: $\text{Re}levância_{Tempo} = \{relevancia_{Tempo} + (\Delta_1 * \Delta_2) * \beta\}$, onde β assume, por padrão, o valor de 0.005 com a finalidade de atenuar o crescimento do documento no *ranking*;
2. SENÃO

2.1. Calcular o quanto o documento será penalizado: $\Delta_1 = 1 - \left(\frac{T_{permanência}(u, d_j, k)}{\left(T_{esperado} / \rho\right)} \right)$;

2.2. Calcular o quanto a relevância do documento poderá decrescer: $\Delta_2 = \frac{relevância_{Tempo}}{2}$;

2.3. Atualizar a Relevância por Tempo: $\mathrm{Re}levância_{Tempo} = \left\{ relevancia_{Tempo} - (\Delta_1 * \Delta_2) * \beta \right\}$, onde β assume, por padrão, o valor de 0.01 e têm como objetivo atenuar a depreciação do documento.

A atualização do modelo, realizada conforme o algoritmo acima, ocorre sempre que o tempo de permanência do usuário no documento for coletado, ou seja, logo após que a sessão do usuário for encerrada (que o documento for fechado). Desta forma, dependendo do tempo obtido, a relevância do documento pode se apreciar ou depreciar, conforme as fórmulas especificadas no algoritmo. A seção seguinte aborda as informações necessárias para a manutenção deste modelo.

3.1.3 Informações necessárias para atualizar o modelo

Para manter o modelo atualizado e, conseqüentemente, utilizar as informações adquiridas no processo de *feedback* de uma consulta, torna-se necessário manter um conjunto de informações em um banco de dados permanente. Desta forma, os algoritmos citados anteriormente necessitam ter um conjunto de informações como entrada, tais como: tempo esperado para a leitura de cada documento, o tempo de permanência neste documento e a relevância por tempo atual, para então gerar as saídas, tais como o *ranking* e a própria realimentação deste banco de dados. Assim, estas informações são processadas e atualizadas na base de consultas. A seguir será apresentado o esquema simplificado desta base de dados.

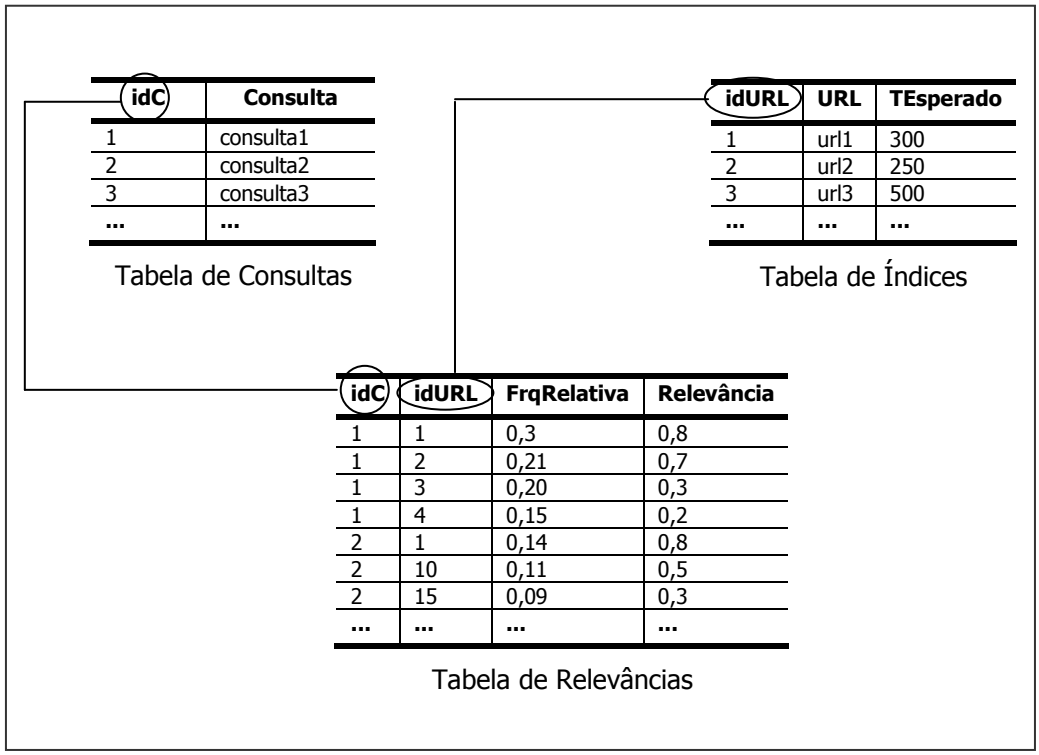

Figura 4 – Esquema simplificado da Base de Consultas

A Figura 4 apresenta um esquema simplificado da base de consultas utilizada pelo mecanismo de busca desenvolvido por DETERS (2003). Nesta base de dados se encontram todas as informações pertinentes para a manutenção do modelo. Neste esquema, a tabela de consultas é a responsável por armazenar todas as consultas realizadas no mecanismo de busca. A tabela de índices fica a representação de cada documento indexado pelo mecanismo de busca. Já a tabela de relevâncias é a responsável por armazenar a relevância por tempo de cada documento à consulta que o retornou, assim como a freqüência relativa obtida pela consulta neste documento. Assim, no momento de atualização desta base de dados, as tabelas que sofrem constantes modificações referem-se às de consulta, pois novas consultas estão constantemente sendo acrescentadas, e a tabela de relevâncias, pois novos registros são incluídos, assim como a constante atualização da relevância por tempo.

4 CLUSTERIZAÇÃO

Clusterização é o processo de agrupar documentos similares (ou relacionados) em categorias, denominadas por *clusters* (BAEZA-YATES & RIBEIRO-NETO, 1999). No entanto, para criar os *clusters* são desenvolvidos métodos que utilizam um conjunto de informações agregadas à coleção de documentos. No caso da recuperação de informação Web, os métodos de clusterização podem utilizar tanto o conteúdo do documento, como também a estrutura de *links* que os interligam (ou os relacionam). Os métodos de clusterização são desenvolvidos levando em consideração características particulares do domínio onde o mesmo será aplicado. Desta forma, privilegia-se a performance da clusterização dentro do domínio ao invés da abrangência do método. A Figura 5 apresenta a clusterização de um conjunto de objetos.

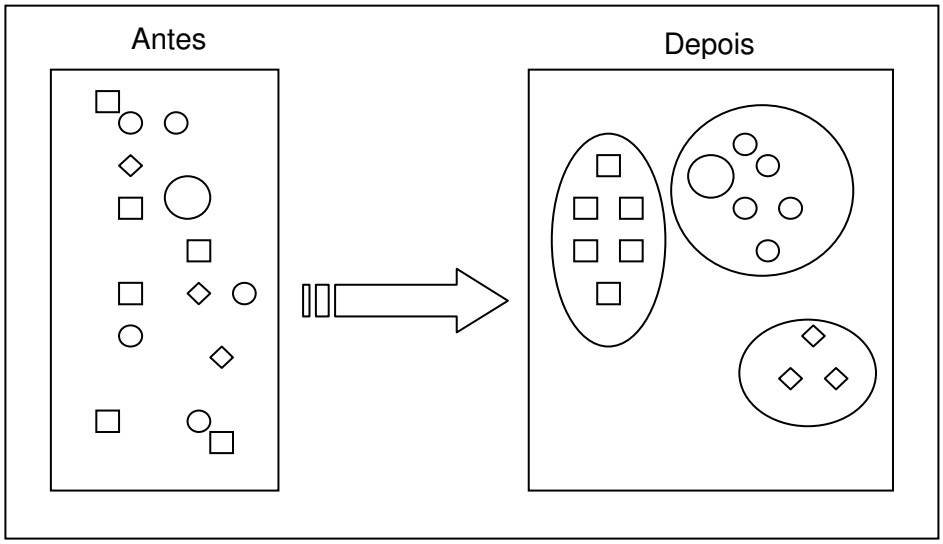

Figura 5 – Definição de *clusters*

No caso apresentado na Figura 5 foi utilizado como critério para a criação dos *clusters* o formato dos objetos. No entanto, outros critérios poderiam ser utilizados, como, por exemplo, a área ocupada ou o tamanho de cada objeto. Por isto é que o método de clusterização é desenvolvido de acordo com as características da coleção analisada.

No entanto, clusterização não é sinônimo de classificação. Apesar de ambos os métodos criarem grupos de documentos, estes dois processos se diferenciam em alguns pontos chaves. Por exemplo, a clusterização de documentos tem por objetivo definir e identificar, de forma automática, grupos de objetos dentro de uma coleção. Já a classificação de documentos visa designar um objeto para um ou vários grupos pré-definido (GAUSSIER & GOUTTE, 2002).

Dentro do contexto da recuperação de informação, a clusterização pode ser utilizada para auxiliar as ferramentas de busca para aprimorar os *rankings* gerados, quer seja criando *clusters* com documentos mais e menos relevantes, quer seja para criar grupos de documentos dentro da lista resultante. Outra possibilidade seria aplicar métodos de clusterização para criar *clusters* na base de índices de um mecanismo de busca para auxiliar a pesquisa por documentos relevantes, diminuindo o volume de documentos a ser pesquisado.

5 METABUSCADOR DESENVOLVIDO

5.1 Considerações Iniciais

O modelo proposto por DETERS (2003) foi validado através do desenvolvimento de um mecanismo de busca, no qual o modelo foi utilizado para gerar os *rankings*. No entanto, para tornar o modelo viável e funcional, foi necessário criar as estruturas que possibilitassem obter e armazenar as informações utilizadas para manter o modelo atualizado. A partir desta necessidade e de algumas adequações feitas para obter estas informações, algumas limitações e problemas no modelo não foram abordadas no referido trabalho. Porém, para que o mesmo possa ser utilizado no meio comercial, estas questões poderão se tornar um empecilho, caso não sejam corretamente sanadas. Dentre estas questões, pode-se citar:

a) Coleta do tempo de permanência do usuário nos documentos: por se tratar de um sistema de busca que trabalha sob um domínio específico, mais precisamente de um tutorial sobre Programação Orientada a Objetos, todas as páginas se encontravam no servidor e os documentos recuperados estariam restritos a esta coleção. Assim, para que o tempo de permanência de cada usuário pudesse ser capturado, cada documento teve sua estrutura alterada para que tal informação pudesse ser recuperada. No entanto, em um sistema aberto que trabalha apenas com os *links* e sua representação, mas não com o documento em si, esta solução não seria viável visto que os documentos não estariam disponíveis.

b) Coleta do tempo esperado de leitura de um documento: como todos os documentos se encontravam no próprio servidor, as estimativas para o tempo

esperado para leitura dos documentos Web foram realizadas durante o processo de indexação desta coleção, contando as palavras e assumindo que um usuário lê três palavras por segundo. Porém, em um ambiente aberto, esta tarefa se torna mais complexa, visto que não se pretende desenvolver toda a estrutura de um mecanismo de busca, mas apenas aplicar o modelo sobre os resultados de um outro sistema de busca, não desenvolvendo, assim, o processo de indexação.

c) Custo computacional envolvido: para manter o modelo, é necessário armazenar e recuperar um grande volume de informações, envolvendo, por isto, elevado custo computacional, principalmente, em termos de processamento e memória. Levando em consideração a quantidade de documentos disponíveis na Web e a quantidade de requisições que uma ferramenta de busca recebe diariamente, o modelo poderia se tornar inviável em um ambiente aberto, por causa da quantidade de recursos necessários para armazenar todas as informações necessárias para manter o modelo atualizado.

Os problemas e limitações apresentados acima tornariam inviável a aplicação deste modelo em qualquer outra ferramenta de pesquisa que trabalhasse em um ambiente aberto e com fins comerciais, tendo em vista as dificuldades de manter e obter as informações necessárias para a atualização do modelo. Assim, para criar a estrutura que atenda as especificações do modelo de ordenação de DETERS (2003) e possibilitar sua utilização em uma ferramenta de busca de domínio aberto e comercial, torna-se necessário definir os meios para administrar e manter as informações a serem utilizadas na atualização das relevâncias de cada documento retornado.

Como o objetivo principal é tornar o modelo de ordenação viável, não se pretende desenvolver todas as estruturas envolvidas no processo de recuperação de informação. Desta forma, foi desenvolvido um metabuscador provido de um mecanismo de ordenação próprio, com a capacidade de reordenar os resultados provenientes de outros buscadores. Porém, o metabuscador aqui desenvolvido não trabalha com resultados provenientes de diversos buscadores, uma vez que apenas se pretende criar a estrutura necessária para validar o modelo de ordenação. Assim, para este trabalho, foi desenvolvido um metabuscador que trabalhe apenas com os resultados fornecidos por um único mecanismo de busca, no caso, os resultados fornecidos pelo Google.

Outra particularidade se refere ao fato de, por se tratar de um metabuscador e, por isto, não indexar os documentos Web, tal como foi realizado no trabalho precursor a este, o modelo de ordenação desenvolvido tem a freqüência relativa das palavras-chave substituída pela ordem com que o mecanismo de busca apresenta seus resultados, pontuando em ordem decrescente cada resultado retornado. A seção seguinte aborda a forma definida para recuperar o tempo de permanência do usuário em cada documento Web retornado, assim como o meio utilizado para obter o tempo esperado para a leitura destes documentos.

5.2 Tempo de Permanência e Tempo Esperado de Leitura

O modelo de ordenação proposto por DETERS (2003) é fundamentado em duas variáveis: o tempo gasto por cada usuário nos documentos Web e o tempo esperado para a leitura destes. Desta forma, por causa dos problemas apresentados na seção 5.1 torna-se necessário desenvolver formas auxiliares para obter estas informações, dadas as particularidades deste trabalho.

5.2.1 Obtendo o tempo esperado para a leitura de um documento

Por se tratar de um metabuscador e, portanto, não implementar as estruturas responsáveis pela indexação dos documentos Web, tornou-se necessário criar um algoritmo para recuperar e calcular o tempo esperado para a leitura de cada documento retornado pela ferramenta de busca. Este algoritmo consiste em:

a) Recuperar o documento Web e fazer uma cópia de todo seu código-fonte para o servidor.

b) Retirar do código-fonte todas as *tags* HTML, deixando apenas o texto do documento.

c) Calcular o tempo esperado para a leitura do documento, partindo, arbitrariamente, da premissa (extremamente conservadora) de que uma pessoa lê três palavras por segundo.

Este algoritmo é executado no momento da atualização da base de consultas, porém, somente se o documento Web ainda não tiver sido indexado (armazenado).

Entretanto, as ferramentas de busca tanto podem recuperar *links* que direcionam para páginas Web, como também que direcionam para outros tipos de documentos, tais como documentos pdf (*Portable Document Format*) ou do tipo texto (doc, txt, etc). Porém, estes documentos, por serem padrões diferentes, não podem ser tratados da mesma forma que os documentos Web. Por isto, assume-se como uma limitação para este trabalho a não indexação (parcial, pois apenas serão armazenados algumas informações, tais como: *links*, data de acesso e o tempo esperado para a leitura deste documento) destes documentos, já que seria necessário criar mecanismos para ler e processar cada um destes formatos.

5.2.2 Obtendo o tempo de permanência dos usuários nos documentos Web

As ferramentas de busca não trabalham com os documentos Web propriamente ditos, mas apenas com uma representação dos mesmos. Assim, quando um sistema de busca retorna uma lista de documentos, o que se é apresentado ao usuário são apenas *links* para sua localização dentro do espaço Web. Por causa desta peculiaridade, não é possível modificar a estrutura do documento de forma a permitir que o tempo de permanência do usuário seja obtido.

Para sanar o problema exposto acima foi utilizado o conceito de *frames* (janelas), onde o documento principal é subdividido em janelas, permitindo, assim, que documentos diferentes sejam abertos nas diferentes janelas criadas, que juntas compõem um único documento a ser apresentado. Desta forma, pode-se abrir, ao mesmo tempo, um documento Web, oriundo da consulta do usuário, e um outro documento contendo a função para coletar o tempo de permanência deste usuário no documento. O tempo passa a ser contabilizado logo que a página for carregada no *browser* do usuário e deixa de ser quando o usuário fechar a página ou selecionar (ou digitar) um outro *link*. O resultado deste processo é retornado e armazenado no servidor do metabuscador para ser posteriormente processado. A seção seguinte aborda a arquitetura do metabuscador desenvolvido, assim como as regras e método criado para gerenciar o conjunto de informações para manter o modelo.

5.3 Arquitetura do Metabuscador e Algoritmo de Pesquisa

A arquitetura apresentada a seguir (Figura 6) representa o metabuscador implementado. Por causa da necessidade de monitorar o usuário e armazenar as informações necessárias para manter o modelo de ordenação, esta arquitetura é composta por um módulo gerenciador e por duas bases de dados (base de consultas e base temporária), além dos módulos comuns a um metabuscador, tais como: mecanismo de consulta e o *ranking*. No caso, a base de consultas é a base de dados utilizada pelo metabuscador e é o resultado do processamento das informações armazenadas na base temporária. Desta forma, qualquer consulta que for submetida ao metabuscador será executada, primeiramente, no Google e, em seguida, será realizada sobre esta base, obtendo, assim, as informações necessárias para a ordenação dos documentos recuperados. O *ranking* resultante será a combinação da ordem como os documentos aparecem no Google com a relevância por tempo armazenada nesta base de dados. Já a base temporária, composta por um conjunto de documentos XML[6], consiste no armazenamento temporário das informações coletadas durante a utilização do metabuscador (lista de URL's recuperadas, tempo de permanência do usuário em cada documento Web, etc) que, posteriormente, serão processadas e atualizará a base de consultas. A Figura 6 apresenta a arquitetura do metabuscador desenvolvido.

[6] *Extensible Markup Language* (XML) é uma meta-linguagem de marcação desenvolvida pelo consórcio W3C (*World Wide Web Consortium*), sendo considerada um padrão para a publicação e transferência de dados de diferentes domínios de aplicação na Web (W3C, 2004).

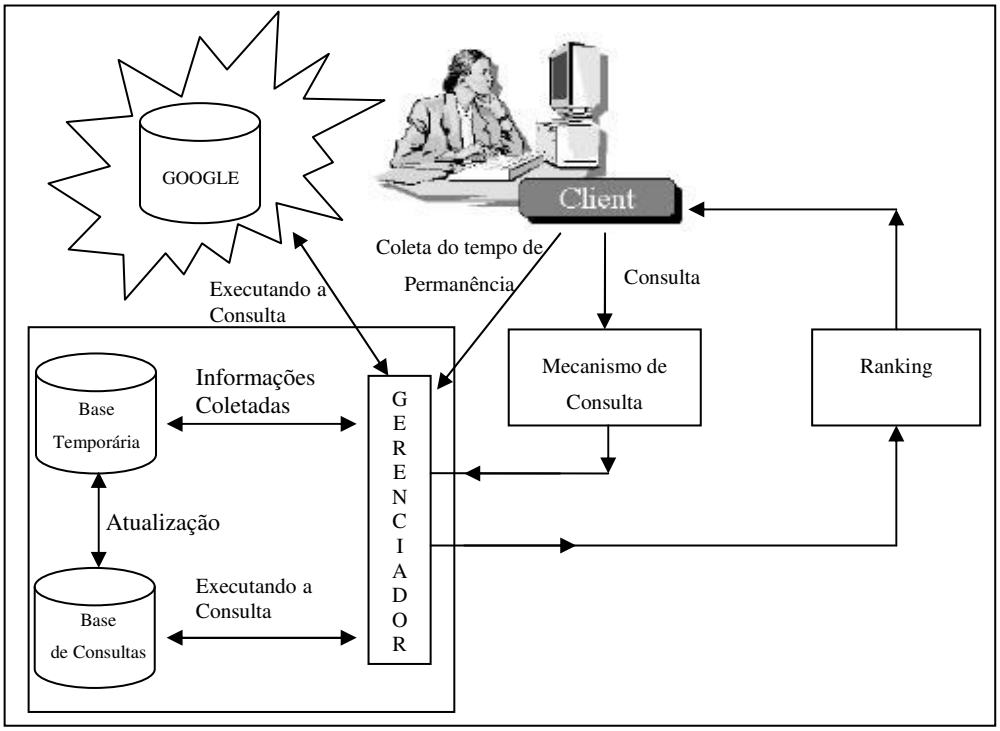

Figura 6 – Arquitetura do metabuscador

Nesta arquitetura são apresentadas todas as estruturas necessárias para prover desde a obtenção e armazenamento das informações relevantes para a manutenção do modelo de ordenação, até o *ranking* resultante a ser apresentado ao usuário. Para tanto, foram criados algoritmos de pesquisa e de atualização das bases de dados, os quais serão apresentados a seguir (Quadro 3).

Quadro 3: Algoritmo de Pesquisa

Entrada: consulta (K)

Procedimentos:

1. Realizar pesquisa no Google;
2. Realizar pesquisa na Base de Consultas;
   2.1. SE **K** não existe na Base de Consultas ENTÃO
      2.1.1. Apresentar os resultados na ordem retornada pelo Google;
   2.2. SENÃO

2.2.1. Ordenar os resultados segundo a combinação da relevância por tempo com a ordem apresentada pelo Google;

3. Coletar as informações (dados da consulta, tempo de permanência do usuário em cada documento Web, URL, etc);
4. Montar o documento XML com as informações da consulta e acrescentá-la na Base Temporária para posterior atualização da Base de Consultas;

O algoritmo de pesquisa (Quadro 3) é implementado no módulo gerenciador. Este módulo é responsável por realizar a pesquisa e ordenar os documentos recuperados, assim como coletar e armazenar as informações necessárias para a manutenção do modelo. Uma vez armazenadas estas informações, torna-se necessário atualizar o modelo, atividade esta desempenhada pelo algoritmo de atualização que será apresentado a seguir (Quadro 4).

Quadro 4: Algoritmo de Atualização

Entrada: Base Temporária

Procedimentos:

1. Recuperar todos os documentos XML;
2. Enquanto houver documento, faça:

2.1. Extrair do documento as informações referente a consulta, documentos e tempo de permanência do usuário em cada documento;

2.2. SE **consulta** já existe ENTÃO;

2.2.1. Atualizar relevância por tempo;

2.3. SENÃO

2.3.1. Verificar se a consulta não está armazenada na base de consultas pouco freqüentes;

2.3.1.1. SE SIM ENTÃO

2.3.1.1.1. Recuperar e reinserir as informações;

2.3.1.1.2. Atualizar a relevância por tempo, dadas as informações adquiridas;

2.3.1.2. SE NÃO ENTÃO

2.3.1.2.1. Inserir informações da consulta;

2.3.1.2.1.1. Inserir consulta;

2.3.1.2.1.2. Calcular o tempo esperado para a leitura do documento;

2.3.1.2.1.3. Calcular e armazenar a relevância por tempo, dadas as informações adquiridas;

2.4. Remover o documento que armazenou temporariamente o conjunto de informações acerca da consulta que foi processada;

Assim, com base na lista de documentos XML, que se encontram armazenadas na base temporária, inicia-se o processo de atualização da base de consultas, onde são mantidas as informações pertinentes para o funcionamento do modelo de ordenação. Porém, o processo de atualização não transcorrerá de forma concorrente ao processo de consulta, visto a dificuldade em se manter o controle de concorrência e pelo custo computacional envolvido. Sendo assim, foram definidas formas para administrar estas bases de dados (consulta e temporária), as quais são apresentadas na seção seguinte.

5.4 Otimização do Custo Computacional

Como descrito anteriormente, para tornar o modelo viável, é necessário prover alternativas para administrar as informações que darão sustentabilidade ao mesmo. Desta forma, foram criadas regras para administrar a base de consultas, assim como um método de clusterização para minimizar o custo computacional envolvida no processo de recuperação e armazenamento destas informações. A forma como a base de consultas é administrada é apresentada nas seções seguintes.

5.4.1 Armazenamento Temporário das Consultas

Uma ferramenta de busca pode receber milhares de consultas diariamente. Essas consultas são processadas e acarretam, por si só, enorme sobrecarga ao Sistema de Gerenciamento de Banco de Dados (SGBD), responsável por gerir a base de dados do sistema de busca. No entanto, para que a ferramenta se mantenha atualizada, outras atividades se fazem necessárias. Destas atividades, duas merecem destaque para a manutenção do metabuscador implementado. A primeira é referente à inclusão das

informações acerca de uma nova consulta realizada. Já a segunda refere-se a atualização das relevâncias de uma consulta reincidente. Como estas atividades seriam realizadas diretamente no banco de dados e como as mesmas seriam executadas concorrentemente às consultas realizadas, estas atividades provocariam uma sobrecarga extra ao SGBD que, conseqüentemente, acarretaria modificações no tempo de resposta ao usuário, devido aos problemas referentes ao controle de concorrência (inserção e consulta de informações).

Pelo motivo exposto acima, optou-se por não executar as tarefas citadas no momento em que elas ocorrerem, mas durante os intervalos em que o metabuscador estiver sendo menos acessado, ou atualizando uma base de consultas espelho à utilizada pelo metabuscador, propagando as atualizações em um momento propício. No entanto, independentemente da vertente adotada, torna-se necessário que as informações sejam armazenadas temporariamente em algum outro formato. No caso, estas informações serão representadas através de documentos XML e armazenadas na base temporária. Este formato foi escolhido por causa das vantagens e facilidades existentes em se criar e manipular estes tipos de documentos, além da possibilidade de se criar um padrão de representação das informações envolvidas, através da definição de DTD's[7]. O DTD, responsável por estruturar e definir quais e em que ordem as informações referentes à consulta deverão aparecer nos documentos XML, será apresentado a seguir (Figura 7).

[7] *Document Type Definition* (DTD) é o responsável por estruturar um documento XML, definindo as partes que um documento poderá ter, assim como descrevendo como elas podem ou não ser usadas, o que pode ser utilizado como conteúdo válido e se são ou não elementos obrigatórios de um documento XML (PITTS-MOULTIS & KIRK, 2000).

```
<!-- DTD responsável por estruturar as informações acerca de uma consulta -->

<!ELEMENT consultaWeb (informacoes, resultados)>
<!ELEMENT informacoes EMPTY>
        <!ATTLIST informacoes
                consulta CDATA #REQUIRED
                tipoPesquisa CDATA #REQUIRED
                dataConsulta CDATA #REQUIRED>
<!ELEMENT resultados (endereco+)
        <!ATTLIST resultados
                fonte CDATA #IMPLIED>
<!ELEMENT endereco EMPTY >
        <!ATTLIST endereco
                URL CDATA #REQUIRED
                tempo_permanencia CDATA #REQUIRED>
```

Figura 7 – DTD que estrutura as informações de um consulta

O DTD apresentado na Figura 7 específica todas as informações importantes e necessárias acerca de uma consulta realizada. Para tanto, foram criados um conjunto de *tags* e atributos pré-definidos para armazenar tais informações, onde os principais são: texto da consulta, a URL, o tempo de permanência do usuário no documento Web e as informações sobre cada item recuperado, tal como a posição do documento no buscador de origem. A seguir (Figura 8) será apresentado um documento XML estruturado a partir do DTD definido.

```
<?xml version="1.0" encoding="utf-16" ?>
  <consultaWeb>
    <informacoes consulta="ASP.NET + C#" tipoPesquisa="" dataConsulta="2005-5-9" />
    <resultados fonte="Google">
      <endereço ID="1" URL="http://www.csharpfriends.com/"
            tempoPermanencia="0" />
      <endereco ID="2" URL="http://www.go-mono.com/" tempoPermanencia="0" />
      <endereco ID="3" URL="http://www.asp-visual-basic-csharp-training.net/"
           tempoPermanencia="0" />
      .
      .
      .
     </resultados>
  </consultaWeb>
```

Figura 8 – Exemplo de um documento XML que representa uma consulta realizada

5.4.2 Eliminação de Índices Inexistentes

A Web é marcada por seu dinamismo, onde documentos são criados e deixam de existir em um curto espaço de tempo. Assim, para evitar que informações referentes a índices que já não existem mais fiquem ocupando espaço na base de consultas, é necessário fazer um constante monitoramento desta base. No entanto, para evitar a eliminação de documentos Web que possam apenas estar inacessível temporariamente (como por exemplo: servidor indisponível), o mesmo só será eliminado na segunda tentativa. Estas verificações são realizadas em períodos pré-definidos como, por exemplo, em intervalos de quinze dias. Este processo será descrito a seguir:

a) Primeira tentativa: caso o item de informação não esteja disponível, o mesmo será marcado como indisponível temporariamente.

b) Segunda tentativa: caso o item permaneça indisponível, o mesmo será excluído da base de consulta, assim como todas as suas relações; caso esteja disponível, o mesmo será remarcado voltando ao estado anterior à primeira tentativa.

No entanto, ao se excluir definitivamente um documento da base de consultas, é necessário, também, excluir todas as relevâncias associadas a este documento e a sua respectiva consulta. Desta forma, evita-se manter informações inúteis na base de consulta, mantendo-a consistente.

5.4.3 Eliminação das Consultas pouco Freqüentes

Para evitar o inchaço da base de consultas é possível, de tempos em tempos, fazer um monitoramento da mesma, eliminando as consultas que são pouco freqüentes. Estas consultas eliminadas caracterizam-se tanto por serem consultas realizadas esporadicamente, como também por serem consultas realizadas com problemas ortográficos que, possivelmente, não seriam repetidas. Porém, as informações referentes a estas consultas eliminadas serão armazenadas, temporariamente, em uma outra base de dados, desenvolvida em XML, para que, caso a consulta seja realizada novamente, a mesma possa ser recuperada, assim como todas suas informações anteriores. O DTD responsável por fazer a estruturação destas informações será apresentado a seguir (Figura 9).

```
<!-- DTD responsável por estruturar as informações acerca de uma consulta que já não ocorre a certo tempo-->

<!ELEMENT consultaWeb (informacoes, resultados)>
<!ELEMENT informacoes EMPTY>
        <!ATTLIST informacoes
                consulta CDATA #REQUIRED
                tipoPesquisa CDATA #REQUIRED
                dataConsulta CDATA #REQUIRED>
<!ELEMENT resultados (endereco+)
        <!ATTLIST resultados
                fonte CDATA #IMPLIED>
<!ELEMENT endereco EMPTY>
        <!ATTLIST endereco
                URL CDATA #REQUIRED
                relevancia_tempo CDATA #REQUIRED>
```

Figura 9 – DTD que estrutura as informações de uma consulta pouco freqüente

O DTD apresentado acima (Figura 9) é semelhante ao apresentado na seção 5.4.1. Porém, neste caso, ao invés de armazenar o tempo de permanência do usuário nos documentos Web, será armazenado a relevância por tempo de cada documento à consulta relacionada. Isto porque esta DTD não estrutura uma nova consulta, mas uma já existente na base de dados e que, por isto, já possui informações a seu respeito. A seguir (Figura 10) será apresentado um documento XML definido segundo o DTD acima.

```
<?xml version="1.0" encoding="utf-16"?>
  <consultaWeb>
    <informacoes consulta="ASP.NET" tipoPesquisa="" dataConsulta="2005-5-9" />
    <resultados fonte="Google">
       <endereco URL="http://www.aspbrasil.com.br" relevancia_tempo="1.5" />
       <endereco URL="http://www.aspnet.com" relevancia_tempo="1.2" />
         .
         .
         .
   </resultados>
</consultaWeb>
```

Figura 10 – Exemplo de um documento XML que representa uma consulta pouco freqüente

Assim, uma vez montada esta base de dados, será necessário fazer verificações sempre que uma consulta tiver que ser inserida na base de consultas do metabuscador. É

importante ressaltar que estas verificações são realizadas no momento em que a base de consultas utilizada pelo metabuscador for atualizada através do processamento das consultas armazenadas na base temporária, e não durante a utilização do metabuscador. As verificações se fazem necessárias para averiguar se a consulta realizada já foi, em algum momento, realizada e retirada da base de dados principal do metabuscador. Caso a mesma seja encontrada, basta re-inserir as informações da consulta na base de dados do metabuscador, com as devidas atualizações.

5.4.4 Método de Clusterização

Como o volume de informações a ser gerido na base de consultas é muito grande e complexo, pretende-se aplicar um método de clusterização, a fim de reduzir o custo computacional envolvido, porém, sem perder informações que poderiam afetar negativamente o modelo de ordenação. O método de clusterização, aqui apresentado, é baseado na criação de *clusters* de consultas, utilizando alguns critérios como, por exemplo, a correlação entre duas ou mais consultas, para compor os *clusters*. Assim, supondo que, para cada consulta, toda a coleção de documentos seja recuperada, e que cabe ao modelo de ordenação a responsabilidade por listar os mais relevantes nas primeiras posições, a base de consultas deste metabuscador poderia ser representada tal como apresentada abaixo (Figura 11).

Lista de Documentos	**Lista de Consultas**				
	C_1	C_2	C_3	...	C_n
D_1	2,5*	2,0	1,0	...	
D_2	1,0	0,9	1,5	...	
D_3	2,0	1,0	0,2	...	
D_4	1,0	0,7	1,0	...	
D_5	0,9	1,0	1,0	...	
D_6	1,1	1,8	1,0	...	
D_7	1,0	1,0	2,8	...	
. . .	. . .	. . .	. . .	...	. . .
D_n				...	
* Relevância por tempo associada ao par documento-consulta					

Figura 11 – Esquema simplificado da base de consultas

O esquema, apresentado acima, representa a base de consultas de forma simplificada. No entanto, aborda todas as informações necessárias para manter o modelo de ordenação, com exceção da relevância do documento atribuída pelo Google, que é obtida quando a consulta é executada na referida ferramenta de busca. Esta base de consultas é composta por informações sobre as consultas, informações sobre os documentos (índices) e informações sobre a relevância por tempo de cada documento a sua consulta correspondente.

Desta forma, o principal problema associado à manutenção desta base de consultas é referente à atualização e recuperação das informações acerca da relevância por tempo de cada documento a sua consulta correspondente. Este problema torna-se mais visível através de simulações. Por exemplo, supondo que sejam realizadas mil (1000) consultas distintas entre si, seria necessário uma quantidade de registros equivalente a mil vezes o número de documentos que compõem a coleção (1000 * nD) para armazenar todas as informações referente a relevância por tempo destes documentos. O problema é que as ferramentas de busca recebem diariamente milhões de consultas, e retornam um número muito maior para cada consulta, o que inviabilizaria o modelo, caso não seja adotada alguma técnica para diminuir a quantidade destas informações.

Assim, com o objetivo de reduzir este custo computacional, optou-se por criar um método de clusterização de consultas, o qual será o responsável por criar *clusters* de consultas, unificando as relevâncias por tempo associadas à lista de resultados de cada consulta. Como cada documento possui uma relevância por tempo associada à consulta que o retornou, é necessário unificá-la, a qual não estará mais associada a uma consulta em específico, mas a todas que fazem parte do *cluster*. Esta nova relevância por tempo será obtida pela média aritmética das relevâncias por tempo deste documento a cada uma das consultas que comporão o *cluster*. Porém, um *cluster* somente poderá ser criado caso as regras apresentadas abaixo sejam satisfeitas:

a) Somente poderão participar de um *cluster* consultas que tenham pelo menos dois documentos em comum nos quais as relevâncias por tempo associadas ao par consulta-documento estejam modificadas (diferentes do valor padrão um), ou seja, consultas que tenham, ao menos, dois documentos lidos por seus usuários.

b) Satisfeita a regra anterior, somente poderão participar de um *cluster* consultas em que exista correlação entre a lista de relevância por tempo associada a cada consulta, ou seja, deve-se ter um padrão entre os resultados apresentados a cada consulta relacionada.

Assim, caso as regras acima sejam satisfeitas, será necessários redefinir os valores das relevâncias por tempo. Para atualizar as relevâncias serão levados em consideração os seguintes critérios: caso os valores das duas relevâncias por tempo sejam diferentes de um (1), ou seja, o documento tenha sido acessado para todas as consultas, a nova relevância será o resultado da média aritmética dos valores antigos; caso o documento não tenha sido acessado para uma das consultas, a nova relevância por tempo será definida pela média dos valores associados aos documentos que foram alterados (lidos); caso nenhum valor tenha sido alterado, será mantido o valor um (1) para a nova relevância. Esta atualização se estenderá a todos os documentos que pertencem às consultas relacionadas ao novo *cluster*.

Como citado anteriormente, um *cluster* somente poderá ser criado caso exista uma proximidade entre as consultas, ou seja, caso haja similaridade entre os valores apresentados para a relevância por tempo dos documentos em comum às consultas relacionadas. Para tanto, será calculada a correlação entre as relevâncias por tempo associadas a cada um dos documentos em comuns das consultas candidatas ao *cluster*. A correlação se caracteriza por ser uma medida padronizada da relação entre duas variáveis, não podendo assumir valor maior do que um (1) ou menor do que menos um (-1). Assim, uma correlação próxima a zero indica que as duas variáveis não estão relacionadas. Uma correlação positiva indica que as duas variáveis movem-se juntas, e a relação é forte quanto mais a correlação se aproxima de um (1). Já uma correlação negativa indica que as duas variáveis movem-se em direções opostas, e que a relação fica mais forte quanto mais próxima de menos um (-1) o valor da correlação for (HOFFMAN, 1998). A correlação entre duas variáveis é dada pela fórmula apresentada na Figura 12.

$$Correlação = \rho_{x,y} = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}}, onde: \overline{X} = \frac{\sum_{i=1}^{n} X_i}{n}, \overline{Y} = \frac{\sum_{i=1}^{n} Y_i}{n}$$

Figura 12 – Cálculo da correlação entre duas variáveis

Como a correlação é calculada para duas variáveis, em caso de um possível *cluster* que envolva mais de duas consultas, será calculada a correlação tomando duas consultas por vez, montando o *cluster* parcial e utilizando o resultado deste *cluster* para calcular a correlação com a próxima consulta e, assim, sucessivamente. Portanto, aplicando as regras sobre a base de consultas apresentada na Figura 11, têm-se as seguintes possibilidades de formação de *clusters*: {(C_1, C_2), (C_1, C_3), (C_2, C_3), (C_1, C_2, C_3)}, onde:

- O possível *cluster* que envolve as consultas C_1 e C_2 satisfaz a primeira regra porque existem pelos menos dois documentos em comum (D_1 e D_6) que foram lidos por seus usuários, ou seja, possuem relevância por tempo diferente de um (C_1D_1 = 2.5, C_1D_6 = 1.1, C_2D_1 = 2.0, C_2D_6 = 1.8). Satisfeita a primeira regra, é necessário calcular a correlação entre as duas consultas para averiguar se o *cluster* pode ser montado, tal como apresentado a seguir:

$$Correlação = \rho_{x,y} = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}} = \frac{0{,}14}{\sqrt{0{,}98}\sqrt{0{,}02}} \cong 1{,}0$$

  Por definição, uma correlação positiva e próxima a um indica que as duas variáveis (consultas) movem-se juntas. Desta forma o *cluster* entre estas duas consultas pode ser criado, como apresentado na Figura 13.

- Os demais *clusters* não poderiam ser criados, pois não satisfazem a primeira regra, ou seja, não possuem pelo menos dois documentos em comum que tenham sido acessados por seus usuários.

O resultado da aplicação deste processo de clusterização sob a base de consulta apresentada anteriormente (Figura 11) está representado através da Figura 13.

Lista de Documentos	Lista de Consultas			
	C_1 C_2	C_3	...	C_n
D_1	2,25*	1,0	...	
D_2	0,9	1,5	...	
D_3	2,0	0,2	...	
D_4	0,7	1,0	...	
D_5	0,9	1,0	...	
D_6	1,45	1,0	...	
D_7	1,0	2,8	...	
. . .	. . .	. . .	...	. . .
D_n			...	
* Relevância por tempo associada ao par documento-consulta				

Figura 13 – Estado final da base de consultas

Desta forma, a base de consultas, à medida que os *clusters* forem sendo criados, tenderá a reduzir o seu tamanho de tal forma que a complexidade C*D tenderá a (C-nc)*D, onde: C = número de consultas; D = número de documentos e nc = número de *clusters*. A seção seguinte apresenta o resultado da avaliação do método de clusterização.

5.5 Avaliação do Método de Clusterização

Foram realizados dois experimentos com o intuito de validar o método de clusterização. Estes experimentos foram realizados aplicando o método de clusterização em uma base de consultas, que foi gerada a partir da coleta de informações oriundas da utilização do metabuscador. Para coletar estas informações, o metabuscador foi disponibilizado para acesso via Web, o qual foi vinculado a um portal acadêmico, o que permitiu um número razoável de acessos diários. Os dados coletados referem-se a um período de dois meses de utilização. Durante este período, foram realizadas, em média, cerca de 100 (cem) consultas diárias, sendo cada uma composta por 10 (dez) documentos recuperados (média). Para cada uma destas consulta foi gerado um documento XML, os quais foram armazenados na base temporária de documentos XML e, posteriormente, processados para atualizar a base de consultas do metabuscador.

Os experimentos são distintos um do outro por causa do valor adotado para definir a proximidade entre duas consultas e, assim, criar o *cluster* relacionado. No primeiro experimento foi utilizado o valor 0.1 para a correlação, ou seja, o *cluster* somente seria criado se a correlação entre as duas consultas fosse superior a 0.1. Já no segundo foi adotado o valor de 0.5. O objetivo de se utilizar valores diferentes para definir se um *cluster* poderá ou não ser criado é verificar até que ponto a redução do custo computacional é válido, se comparado com as distorções que o método de clusterização poderá gerar ao modelo. Por isto, para cada experimento, foi calculado o erro gerado pelo método de clusterização, se comparado à base de consultas sem que o mesmo fosse aplicado.

No entanto, para evitar a criação de *clusters* com documentos que já deixaram de existir, a base de consultas foi constantemente monitorada, retirando desta todas as informações que estivessem relacionadas a documentos que deixaram de existir na Web. Outro ponto abordado refere-se à verificação e eliminação das consultas pouco freqüentes da base de dados utilizada pelo metabuscador. Esta segunda verificação pretende eliminar consultas que não iriam se repetir, ou porque não foram encontrados documentos relevantes para a consulta, ou por causa da forma como a mesma foi elaborada como, por exemplo, consultas com problemas ortográficos. Esta verificação é necessária, porém, também era preciso evitar que consultas que sejam utilizadas apenas em períodos pré-definidos, tal como, informações acerca de um congresso que é realizado de ano em ano, sejam excluídas definitivamente da base de consultas. Assim, optou-se por não perder as informações existentes até o momento em que a mesma for retirada da base de consultas, tal como a relevância por tempo dos documentos a suas respectivas consultas. Estas informações foram retiradas da base de consultas, mas foram armazenadas em uma outra base de dados desenvolvida em XML, possibilitando com que estas informações sejam recuperadas, caso as mesmas ocorram em algum outro momento durante a utilização do metabuscador, tal como apresentado na seção 5.4.3. A seguir serão apresentados os resultados obtidos com o processo de clusterização.

5.5.1 Resultados Obtidos pelo Método de Clusterização

Como já foi ressaltado, o objetivo do método de clusterização é reduzir o espaço utilizado para armazenar os registros que contém as informações referentes a relevância

por tempo de cada documento à consulta que o retornou. Desta forma, o método de clusterização criado, por necessitar da relevância por tempo associada ao par documento-consulta, somente teria sua eficácia validada após um período razoável de utilização do metabuscador para a coleta de informações. Assim, durante o período de validação, o método de clusterização foi aplicado seis vezes, em intervalos de quinze dias. Os experimentos foram aplicados no mesmo conjunto de dados, sendo que isto pode ser verificado analisando a quantidade de registros que foram acrescentados de uma clusterização para outra. Por exemplo, utilizando os dados da terceira e da quarta clusterização do primeiro experimento (Tabela 1) tem-se: 33.842[8] – 20.525[9] = 13.317 registros acrescentados. Este mesmo valor será obtido se o cálculo correspondente for realizado levando em consideração os dados equivalentes no segundo experimento. Portanto, o que difere um experimento do outro é o valor utilizado para correlação, assim como o número de registros reduzidos na base de consulta por causa do número de *clusters* criados. Os resultados serão apresentados na tabela seguinte (Tabela 1 e 2).

Tabela 1 – Primeiro Experimento: correlação igual ou superior a 0.1

Clusterização	**Estado da Base de Consultas (nº de registros)**					
	Primeira	Segunda	Terceira	Quarta	Quinta	Sexta
Antes da clusterização	8846	18825	24895	33842	42121	50594
Após da clusterização	8115	17180	20525	26351	32798	39312
Nº *clusters* criados	1	17	95	173	359	627

Tabela 2 – Segundo Experimento: correlação igual ou superior a 0.5

Clusterização	**Estado da Base de Consultas (nº de registros)**					
	Primeira	Segunda	Terceira	Quarta	Quinta	Sexta
Antes da clusterização	8846	18825	25726	37440	48795	61750
Após da clusterização	8115	18011	24123	33025	43954	52084
Nº *clusters* criados	1	9	45	109	215	372

[8] Quantidade de registros existente antes da quarta clusterização.

[9] Quantidade de registros existente após a terceira clusterização.

Analisando os resultados apresentados acima (Tabela 1 e 2), nota-se que, à medida que o valor para a correlação aumenta, o número de *clusters* diminui, reduzindo também a eficácia do método de clusterização no que diz respeito à redução da quantidade de informação armazenada. Isto se justifica por causa das características atribuídas ao método, que exige que as consultas relacionadas tenham pelo menos dois documentos em comum acessados (valor da relevância por tempo diferente de um) e que tenham similaridade entre os valores apresentados pela relevância por tempo. Assim, quanto maior o valor para a correlação, maior a proximidade exigida entre os valores apresentados para a relevância por tempo associado ao par consulta-documento para que seja criado o *cluster*.

Por isto, para averiguar se a significativa redução do primeiro experimento (correlação igual ou superior a 0.1) não provocou distorções às informações pertinentes ao modelo de ordenação será calculado o erro gerado pelo método de clusterização. Este erro será definido calculando a média dos desvios padrão existente entre os valores da relevância por tempo de cada documento à sua consulta de origem com os valores finais apresentados no *cluster*. O desvio padrão se caracteriza por assumir apenas valores positivos e quanto maior for, maior será a dispersão dos dados e, quanto mais próximo de zero, menor a dispersão. A fórmula para se calcular o desvio padrão será apresentada abaixo:

$$s = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \overline{X})^2}{n-1}}, onde: \overline{X} = \frac{\sum_{i=1}^{n} X_i}{n}$$

Para exemplificar o cálculo do erro, será utilizado o seguinte caso (Figura 14):

Lista de Documentos	**Lista de Consultas**		
	Consulta 1	Consulta 2	*Cluster*
D_1	2,5	2,0	2,25
D_2	1,1	1,1	1,1
D_3	2,0	1,0	2,0

Figura 14 – Base exemplo utilizada para o cálculo do erro associado ao método de clusterização

Calculando o desvio padrão de cada linha têm-se os seguintes resultados: 0.25, 0.0, 0.57. O erro associado ao processo de clusterização do conjunto de dados apresentados na Figura 14 seria, portanto, igual a 0.27, ou seja, a média dos desvios padrões obtidos em cada linha que teve o valor da relevância por tempo alterado.

Seguindo o mesmo raciocínio apresentado acima, foi calculado o erro gerado pelo método de clusterização para os dois experimentos, se comparado com a base de dados que não sofreu a clusterização. O primeiro experimento, o qual foi realizado levando em consideração a correlação superior a 0.1, o erro associado foi de aproximadamente 0.39. Já para o segundo (correlação superior a 0.5), o erro foi de aproximadamente 0.095. A distorção provocada pelo primeiro experimento foi significativa, caso seja comparada ao intervalo de variação da relevância por tempo, definido entre zero (0) e três (3). Porém, a redução da base de dados foi bem mais significativa.

5.5.2 Considerações sobre os resultados obtidos

O objetivo do método de clusterização é reduzir o número de informações armazenadas na base de consultas, sem, contudo, perder informações relevantes para a manutenção do modelo de ordenação adotado. Desta forma, analisando os resultados apresentados na seção 6.5.1, pode-se fazer as seguintes considerações sobre a eficácia do método criado:

a) O número de registros da base de consultas tenderá a diminuir à medida que houver a reincidência das consultas, porém, esta redução ficará limitada ao número de documentos que a coleção contiver. O limite inferior será alcançado apenas se todas as consultas forem similares, o que ocasionaria a criação de um único *cluster*, que conteria todas as consultas.

b) O método de clusterização provoca distorções ao modelo de ordenação, uma vez que, no momento de criar um *cluster*, a relevância por tempo deve ser unificada, não mais estando associada ao par consulta-documento, mas ao *cluster*-documento. Por isto, deve-se definir um valor para a correlação entre as consultas de forma a não possibilitar a criação de *clusters* contendo consultas

com pouca proximidade entre seus resultados, assim como não limitar muito o processo de clusterização.

c) Analisando as consultas realizadas (através dos documentos XML que as representam), nota-se que, para muitas consultas, apenas um documento da lista retornada foi acessado, ou seja, lido. Isto porque, em muitas situações, o documento relevante para a consulta do usuário está na primeira posição, ou o mesmo se satisfaz com o documento escolhido. Por causa deste fato, muitos *clusters* deixaram de ser criados devido a uma das restrições, que define que um *cluster* somente poderia ser criado caso houvesse nas consultas candidatas dois documentos em comum que tivessem sido acessados (lidos).

5.5.3 Formalização da Redução

Apesar da redução proporcionada com a utilização das regras e do método de clusterização, a base de consultas tenderá a crescer no decorrer do tempo, porém, com uma taxa de crescimento menor do que a apresentada caso o método de clusterização não fosse utilizado. Para efeito de comparação foi acompanhado o crescimento de três bases de consultas, uma da forma como desenvolvida no trabalho de DETERS (2003) e as outras duas aplicando o método e as verificações criadas neste trabalho. Os gráficos resultantes desta comparação são apresentados através da figura a seguir (Figura 15).

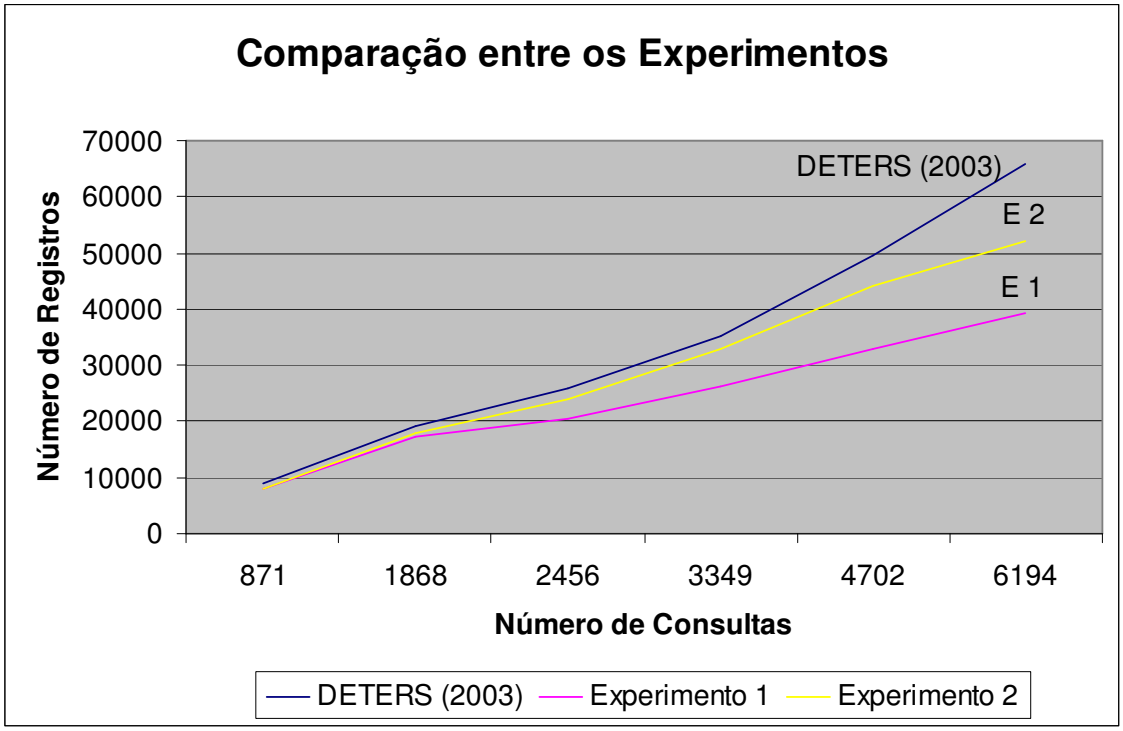

Figura 15 – Comparação entre os experimentos

Analisando as curvas apresentadas nos gráficos acima, constata-se que o número de registros existente na base de consultas que não utiliza o método de clusterização cresce a uma taxa bem superior às bases de dados que o usam. Assim, formalizando esta redução, tem-se: supondo uma coleção contendo 'D' documentos, que sejam identificados por 'n' palavras-chaves cada um. Supondo ainda que, para cada consulta, toda a coleção seja recuperada, ficando a responsabilidade para o modelo de ordenação definir quais destes são mais relevantes para a consulta realizada, e que 'C' consultas tivessem sido executadas, necessitando de um número 'C*D' de registros para armazenar todas as informações pertinentes ao modelo.

Porém, sabendo que o número de documentos na coleção é um valor fixo, o problema é identificar e reduzir ao máximo o valor atribuído a 'C', ou seja, reduzir o número de consultas armazenadas. Como cada documento pode ser recuperado pela combinação de suas palavras-chaves e cada uma destas combinações representa uma consulta diferente, o número de possíveis consultas que surgiriam a partir de cada documento seria algo em torno de:

$$\text{Documento } D_1\text{: } [np = 2] \rightarrow (p_1, p_2) = \{\{p_1\}, \{p_2\}, \{p_1, p_2\}\} \rightarrow c = 3;$$

Documento D_2: [np = 3] → (p_1, p_2, p_3) = {{p_1}, {p_2}, {p_3}, {p_1, p_2}, {p_1, p_3}, {p_2, p_3}, {p_1, p_2, p_3}} → c = 7;

Documento D*n*: [np = n] → (p_1, p_2,..., p_n) = {{p_1}, {p_2},..., {p_n}, {p_1, p_2},..., {p_1, p_n},..., {p_1, p_2,..., p_n}} → c = $\frac{n!}{1!(n-1)!} + \frac{n!}{2!(n-2)!} + \ldots + \frac{n!}{n!(n-n)!} \rightarrow$ c = $\sum_{k=1}^{n} \frac{n!}{k!(n-k)!}$, ou seja, somatório da combinação simples de *n* elementos tomados *k* a *k*.

Desta forma, a ferramenta de busca desenvolvida em DETERS (2003) necessitaria de um número de registros equivalente ao somatório de todas as possibilidades de consultas vezes o número de documentos da coleção para armazenar todas as possibilidades de relevância por tempo de um documento a uma consulta, como apresentado a seguir:

$$N_{registros} = D * C \text{ , onde: C} = \sum_{i=1}^{D} c_i \text{ e } c_i = \sum_{k=1}^{n} \frac{n!}{k!(n-k)!}$$

Esta grande quantidade de registros necessários para armazenar a relevância por tempo tornaria inviável a utilização do modelo de ordenação para uma ferramenta de busca na qual o número de documentos indexados é extremamente elevado. Isto porque um mesmo documento poderia ser retornado em qualquer uma das combinações de suas palavras-chave. Assim, da forma como a ferramenta de busca foi desenvolvida em DETERS (2003), o número de registros para armazenar a relevância por tempo de cada documento à sua consulta, pode ser generalizada através da seguinte expressão:

$$N_{registros} = D \xrightarrow{tendendo} D * \sum_{i=1}^{D} c_i \text{ , onde c} = \sum_{k=1}^{n} \frac{n!}{k!(n-k)!}$$

Ou seja, caso haja apenas uma consulta realizada, o número de registros seria equivalente ao número de documentos existentes na coleção. Porém, à medida que as consultas forem sendo realizadas, este número tenderá ao somatório das possíveis consultas a serem realizadas vezes o número de documentos da coleção.

O método de clusterização desenvolvido tem como objetivo atenuar o problema descrito acima, criando *clusters* para representar consultas similares. Desta forma, poderia ser criado um único *cluster* contendo todas as possíveis consultas oriundas da combinação de palavras-chave de seus documentos, fazendo com que o número de registros necessários para armazenar a relevância por tempo de cada documento à sua consulta tenda ao número de documentos indexados na base de consulta, tal como:

$$N_{registros} = D * \sum_{i=0}^{D} c_i \xrightarrow{tendendo} D \text{, onde r} = \sum_{k=1}^{n} \frac{n!}{k!(n-k)!}$$

A fórmula apresentada acima é o inverso da apresentada para o trabalho de DETERS (2003), onde a base de relevâncias tende a se aproximar do número de documentos da coleção. Assim, à medida que os clusters forem se formando, o valor para o *C* (número de consultas) se reduzirá e, conseqüentemente, o número de registros necessários para armazenar as informações pertinentes ao modelo, também. A seguir serão apresentadas as considerações finais sobre este trabalho.

6 CONSIDERAÇÕES FINAIS

O modelo de ordenação proposto por DETERS (2003) agrega informações obtidas do usuário, no caso, o tempo de permanência deste nos documentos recuperados, ao processo de organização dos resultados obtidos pela ferramenta de busca. Desta forma, a ênfase é atribuída ao *ranking*, que será modificado no decorrer da utilização e da repetição das consultas realizadas, uma vez que as informações obtidas são utilizadas no processo de *feedback* de uma consulta. Porém, para manter este modelo, inúmeras informações devem ser armazenadas, como, por exemplo: as informações sobre as consultas realizadas, os documentos recuperados e a relevância por tempo de cada documento a consulta que o retornou. O problema é que o custo computacional envolvido para armazenar (e recuperar) estas informações é relativamente alto.

Esta dissertação abordou e buscou solucionar as lacunas deixadas no trabalho desenvolvido por DETERS (2003), priorizando o custo computacional envolvido para manter o modelo de ordenação em uma ferramenta de busca comercial e trabalhe com a recuperação de informações de qualquer tipo de informação (domínio aberto). Para tanto, foi desenvolvido um metabuscador, responsável por recuperar informações em outras ferramentas de busca que, no caso, se resumiu aos resultados fornecidos pelo Google, assim como as estruturas necessárias para manter e recuperar as informações pertinentes para a manutenção do modelo de ordenação.

O método de clusterização proporcionou uma redução significativa da base de consultas testada. No entanto, não é possível afirmar que a taxa de crescimento será muito inferior à apresentada pela base de consultas desenvolvida em DETERS (2003). Isto porque a redução irá depender do nível de utilização da ferramenta de busca a qual

o modelo foi adotado e do comportamento do usuário ao analisar os documentos gerados. Assim, não basta que as consultas sejam realizadas, é necessário que o usuário acesse pelo menos dois documentos para que os *clusters* sejam criados e, conseqüentemente, o número de registros seja reduzido. De qualquer forma, na pior das hipóteses, a espaço necessário para armazenar todas as informações pertinentes ao modelo será igual ao apresentado em DETERS, ou seja, caso nenhum *cluster* tenha sido criado. Porém, à medida que as consultas forem ocorrendo e os documentos forem avaliados, esta base de consultas, tenderá a ser reduzida, diminuindo gradualmente o número de informações armazenadas. Esta redução, em situações que envolvam diversas consultas similares e que contenham inúmeros documentos bem avaliados para estas consultas, poderá ser bem significativa o que proporcionaria a utilização do modelo.

Analisando os dados das consultas realizadas, nota-se que muitos *clusters* deixaram de ser criados devido à restrição definida por uma das regras, a qual estipula que o *cluster* somente poderia ser criado caso dois documentos comuns às consultas relacionadas tenham sido acessados, ou seja, lidos. Isto se verifica porque os usuários, em muitas situações, acessam somente um documento da lista de resultados, ou porque se satisfez com o conteúdo do mesmo, ou porque a consulta foi mal formulada e não trouxe os resultados que atendam a sua necessidade. No entanto, como esta regra é necessária para que se tenha um número mínimo de documento em comum e para que haja, assim, uma redução significativa do número de registros no momento da criação do *cluster*, além de ser uma condição necessária para que possa ser calculada a correlação entre os resultados das consultas candidatas, pode-se, nos experimentos futuros, averiguar a possibilidade de tornar esta regra mais flexível. Assim, para estes casos, se exigiria apenas que existissem dois documentos em comum, mas não se exigiria que os mesmos tivessem sido lidos para todas as consultas candidatas ao *cluster*.

Outra importante observação diz respeito ao valor utilizado para a correlação entre os resultados das consultas candidatas de um *cluster*, pois este parâmetro interfere diretamente na eficiência e na qualidade do processo de clusterização. Desta forma, definir um valor elevado para a correlação pode ocasionar uma menor distorção ao

modelo de ordenação, aumentando a qualidade dos *clusters* criados. No entanto, a redução do volume de informações tenderá a ser menos significativo, o que reduzirá a eficácia do processo de clusterização. Por outro lado, utilizando um valor menor, a ordem se inverterá, pois se ganha em eficácia e perde-se em qualidade dos *clusters*. Assim, a melhor alternativa seria utilizar um valor intermediário para a correlação, privilegiando tanto a qualidade, como a eficácia do método de clusterização.

Com os resultados obtidos neste trabalho pode-se verificar uma melhoria na gerência das informações necessárias para a manutenção do modelo de ordenação proposto por DETERS (2003). Porém, este estudo representa uma pequena parte do que é necessário para implantar o modelo em qualquer ferramenta de busca, visto as características de cada sistema de Recuperação de Informação Web. No entanto, isto não impossibilita que se chegue a algumas conclusões sobre a possibilidade da implantação do modelo nestas ferramentas. Por exemplo, para um metabuscador, tal como o utilizado neste trabalho, é viável sua utilização. Isto porque a única base de dados a ser gerenciada pelo sistema é a própria na qual estão armazenadas as informações pertinentes ao modelo, além do mais, como apresentado neste trabalho, o método de clusterização consegue minimizar o crescimento desta base de dados. Já para os diretórios e mecanismos de busca sua utilização torna-se mais complexa, pois, além da base de índices, estas ferramentas teriam que gerenciar a base de dados do modelo, que, possivelmente, seria tão volumosa quanto a primeira, sendo necessário, por isto, novos estudos para viabilizar a utilização deste modelo de ordenação, sem comprometer a velocidade da recuperação.

Outro ponto importante refere-se aos problemas existentes para se obter o tempo de permanência do usuário nos documentos Web, assim como para obter o tempo esperado para a leitura de um determinado documento. A primeira situação seria um problema comum para todas as classes de ferramentas de busca. Já a segunda, seria um problema maior para as ferramentas que não indexam os documentos Web (como, por exemplo, metabuscadores ou diretórios), visto que esta tarefa já é parcialmente realizada pelos mecanismos de busca, no momento que os mesmos constroem a representação destes documentos. Por causa destes problemas, além do excesso de informações a ser gerenciada, outra possibilidade para a utilização do modelo de ordenação proposto por

DETERS (2003) seria sua implantação em ferramentas de pesquisas internas de servidores (por exemplo, de hospedagem) ou de organizações. Desta forma, o volume de informações a serem pesquisas seria reduzida e delimitada, o que facilitaria o processo de gerência das informações necessárias para manter o modelo.

6.1 Trabalhos Futuros

Durante o desenvolvimento deste trabalho, alguns pontos importantes não foram considerados, lacunas estas que podem ser abordadas em trabalhos futuros. Uma destas lacunas refere-se a realização de um estudo mais aprofundado na definição do tempo esperado para a leitura de um determinado documento Web. Nesta pesquisa, poderiam ser desenvolvidos métodos para tentar prever o tempo gasto para interpretar uma figura, assim como o tempo previsto para ler um documento que esteja em outros formatos (pdf, doc, ppt, etc).

Como o objetivo deste trabalho foi criar os meios necessários para tornar o modelo de ordenação desenvolvido por DETERS (2003) viável para ser implantando em uma ferramenta de busca genérica, não foi realizado uma análise dos *rankings* gerados. Por isto, faz-se necessário desenvolver ou utilizar uma metodologia de avaliação para avaliar os resultados gerados e, assim, verificar o quanto o modelo de ordenação contribuiu para aumentar a qualidade dos resultados oferecidos pelo sistema de busca desenvolvido.

7 REFERÊNCIAS BIBLIOGRÁFICAS

(ARASU, 2001) ARASU, Arvind et al. Searching the Web. **ACM Transactions on Internet Technology (TOIT)**, v. 1, n. 1, p. 2-43. New York: ACM Press, 2001.

(BAEZA-YATES & RIBEIRO-NETO, 1999) BAEZA-YATES, R., RIBEIRO-NETO, B. **Modern Information Retrieval**. New York: ACM Press, Addison Wesley, 1999. 513p.

(BERNERS-LEE, 1994) BERNERS-LEE, Tim et al. The World-Wide Web. **Communications of the ACM**, v. 37, n. 8, p.76-82. . New York: ACM Press, 1994.

(BOCK, 2001) BOCK, Lia. Só vendo para crer: Técnica de leitura rápida usa um método chamado fisioterapia ocular; ele é contestado por especialistas. **Istoé online**, 08 ago. 2001. Disponível em: <http://www.terra.com.br/istoe/1662/medicina/1662_so_vendo_pra_crer.htm>. Acesso em: 07 jun. 2005.

(DETERS, 2003) DETERS, Janice Inês. **Método de Ordenação de Documentos na Web Baseado no Tempo de Permanência**. Florianópolis, 2003. 88 f. Dissertação (Mestrado em Ciências da Computação) - Universidade Federal de Santa Catarina, Florianópolis, 2003.

(GAUSSIER & GOUTTE, 2002) GAUSSIER, Eric, GOUTTE, Cyril. **Probabilistic Models for Hierarchical Clustering and Categorization: Applications in the Information Society**. 2002. Disponível em: <http://citeseer.ist.psu.edu/714738.html >. Acesso em: 28 de fev 2005.

(JOACHIMS, 2002) JOACHIMS, Thorsten. Optimizing Search Engines Using Clickthrough Data. In: Conference on Knowledge Discovery in Data. **Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**, p. 133-142. New York: ACM Press, 2002.

(HOFFMAN, 1998) HOFFMAN, Rodolfo. **Estatística para Economistas**. São Paulo: Pioneira de Ciências Sociais, 1998.

(HUIBERS & LAMAS & RIJSBERGEN, 1996) HUIBERS, T.W.C., LALMAS, M., RIJSBERGEN, C.J. van. **Information Retrieval and Situation Theory**. ACM SIGIR Forum, v. 30, n. 1, p 11-25. New York: ACM Press, 1996.

(KAMVAR, 2003) KAMVAR, Sepandar D. et al. Extrapolation Methods for Accelerating PageRank Computations. In: International World Wide Web Conference. **Proceedings of the twelfth international conference on World Wide Web**, p. 261-270. New York: ACM Press, 2003.

(KOBAYASHI & TAKEDA, 2000) KOBAYASHI, Mei, TAKEDA, Koichi. **Information Retrieval on the Web**. ACM Computing Surveys (CSUR), v. 32, n. 2, p. 144-173. New York: ACM Press, 2000.

(MACEDO, 2001) MACEDO, Joaquim. **Recuperação de Informação Textual Distribuída por Fontes Autônomas com Sobreposição**. Braga, 2001. 275 f. Tese (Doutorado em Informática) – Universidade do Minho, Braga, 2001.

(MONTAGUE & ASLAM, 2002) MONTAGUE, Mark, ASLAM, Javed A. Condorcet Fusion for Improved Retrieval. **Proceedings of the 11 International Conf. on Information Knowledge and Management (CIKM)**, p. 538-548. New York: ACM Press, 2002.

(MOOERS, 1951) MOOERS, C. **Zatocoding applied to mechanical organization of Knowledge**. American Documentation, v. 2, n. 1, p. 20-32, 1951.

(MOSTAFA, 2002) MOSTAFA, Javed. How do Internet search engines work?. **Scientific American: Ask the Experts**. out. 2002. Disponível em:

<http://www.sciam.com/askexpert_question.cfm?articleID=00009562-3FFB-1DA4-815A809EC5880000&catID=3>. Acesso em: 22 set. 2003.

(PITTS-MOULTIS & KIRK, 2000) PITTS-MOULTIS, Natanya. KIRK, Cheryl. **XML: Black Book**. São Paulo: Makron Books, 2000. 627p.

(RIJSBERGEN, 1979) RIJSBERGEN, C. J. van. **Information Retrieval**. Londres: Butterworths, 2ª ed, 1979. Disponível em: <http://citeseer.nj.nec.com/vanrijsbergen79information.html>. Acesso em: 07 nov. 2003.

(SARACEVIC, 1995) SARACEVIC, T. Interdisciplinary Nature of Information Science. **Ciência da Informação**, Brasília, v. 24, n.1. p. 31-36, 1995.

(SMITH & HURSON, 2003) LaQuiesia S., SMITH. Ali R. HURSON. A Search Engine Selection Methodology. In: Information Technology: Coding and Computing. **Proceedings of the International Conference on Information Technology: Computers and Communications (ITCC'03)**, p. 122 -129. IEEE, 2003.

(W3C, 2004) World Wide Web Consortium (W3C). **Extensible Markup Language (XML) 1.1**. fev. 2004. Disponível em: <http://www.w3.org/TR/2004/REC-xml11-20040204/>. Acesso em: 15 fev. 2005.

8 ANEXOS

Exemplos de alguns *clusters* criados

1º Exemplo	cursos do CEULP/ULBRA
	cursos de graduacao do ceulp
	sistemas de informação do CEULP/ULBRA
	sistemas de informação + CEULP
	notícias do curso de Sistemas de Informação
	professores dos cursos + CEULP
	Sistemas de Informação do CEULP/ULBRA
	Área do curso de sistema do CEULP
	professores de sistemas de informação + ceulp
	monografias de sistemas de informação do CEULP/ULBRA
	encoinfo + CEULP/ULBRA
	Resultados + vestibular + ceulp
	Vestibular + ceulp
	resultados do vestibular do curso de sistemas
	resultados do vestibular de sistemas de informação
	Vestibular + resultados + fisioterapia
	resultados + fisioterapia

2º Exemplo	Linguagens de programação C# Linguagem C# Programação Web ASP.NET e C# programação ASP .NET C# programando com C# programação com asp.net PHP + C# + JSP Asp.net programação dinâmica com c# php jsp jsp + programação web
3º Exemplo	prouni resultado do prouni prouni + ceulp/ulbra resultado do prouni para o ceulp/ulbra universidade para todos + palmas resultado do prouni + ceulp/ulbra informações sobre o prouni o que é o prouni? para que serve o prouni lista do prouni
4º Exemplo	“e-book voltado para administração de empresas” e-book + administração de empresas tutorial para administração de empresas apostilas de administração de empresas
5º Exemplo	normas ABNT como se utiliza as normas abnt

	exemplos de referências nas normas ABNT normas + abnt como se referencia nas normas abnt modelos + normas + abnt trabalhos científicos abnt
6º Exemplo	smalltalk smalltalk + orientação a objetos programação orientada a objetos para que serve a orientação de objetos poo Java e poo recursos de poo c++ características da orientação a objetos polimorfismo o que é polimorfimo?
7º Exemplo	células-tronco células tronco e biologia projetos que envolvem células-tronco “pesquisas sobre células-tronco” pesquisas sobre células-tronco pesquisas + células-tronco benefícios das pesquisas sobre células-tronco resultados de pesquisas de células-tronco